Mandriva 2007 » Fedora Core 6 » Tcl

LXF DVD внутри!

Mandriva 2007
Fedora Core 6

LINUX FORMAT

Главное в мире Linux

Январь 2007 № 1 (87/88)

Сделайте это в
MONO
Новая серия для начинающих!
с. 56

Вкус будущего!
KDE 4

Узнайте, как будет выглядеть рабочий стол завтрашнего дня
- » Мнения разработчиков
- » Новые возможности
- » Исходные тексты на DVD

Blender
Моделируйте вместе с нами с. 90

Tcl
и другие экзотические языки с. 38

Рассмотрим и оценим:



«Я уже было встал и ушел – мне казалось, что глупее ничего быть не может»

Джефф Во об истоках Ubuntu с. 24

4 607130 820476





К Вашим услугам...

Как известно, релиз Fedora Core 6 был задержан – и поэтому не попал на DVD декабрьского номера. Вот мы и решили спросить у Команды LXF: «А каким было лучше оправдание вашего опоздания?»



Пол Хадсон
Я всегда высылаю результаты вовремя, но иногда они задерживаются в пути. Интернет – это не экспресс, это скопище проводов!



Грэм Моррисон
Мне всегда говорили, что дедлайн – это кривая Безье, вид сбоку.



Майк Сондерс
Извините, что задержал HotPicks – мой ноутбук взорвался и мне пришлось набивать текст на Casio Data Bank.



Эфрейн Эрнандес-Мендоса
Я никогда не опаздываю – но однажды упустил свой корабль, ибо забыл выключить чайник.



Ребекка Смолли
Простите, янеложиласьвсюночьпринялаcлишкоммноговазвеселилаcьизаснулав библиотекечтоэтотутувасзаогонькимигают?



Эндрью Грегори
Мое лучшее оправдание: «Я заблудился и уехал в Бирмингем», потому что это всегда срабатывает.



Нейл Ботвик
Однажды я опоздал на поезд из-за листьев на проводах. Правда, они еще не успели опасть со своих деревьев.



Дэвид Картрайт
Не вешайте всех собак на меня – редактор послал письмо в какой-то левый журнал и мне постоянно за это мстят.



Энди Ченнел
Я честен до безобразия и обычно говорю: «Вчерашний день я провел в магазине музтоваров».



Ричард Коббет
Я как-то слышал: «На кухне прорвало трубу» от того же фрилансера, что минутой раньше сказал: «Моя кошка попала под машину»



Энди Хадсон
Простите, мой сын решил смазать клавиатуру клубничным джемом и клавиша Backspace запала.

Миссия журнала

Corporate Patron Free Software Foundation

- Пропаганда свободного ПО в России
- Продвижение решений с открытым кодом в бизнес-сообществе
- Поддержка российского Open Source сообщества
- Организация трибуны для разработчиков свободного ПО
- Обратная связь между разработчиками и потребителями ПО



О сходстве KDE и огров

Нет, я не стану проводить аналогии между KDE и неплохой 3D-библиотекой Ogre, которую мы только-только закончили изучать – так что вы можете сделать это самостоятельно. Я имею в виду настоящих огров – больших, страшных, обретших всемирную популярность благодаря Шреку.

Перефразируя этого самого Шрека, можно сказать: «У огров есть слои, и у KDE есть слои». Причем слоев этих становится с каждым днем все больше и больше. KDE4 обещает быть прекрасным рабочим окружением, но какой ценой? Каждый новомодный компонент, получивший звучное кодовое имя, является, по сути, оберткой: Phonon – вокруг Xine/NMM, Solid – вокруг HAL/NetworkManager (которые сами по себе являются прослойками, причем совсем не тонкими). Где уж тут говорить об изяществе?

Не подумайте, что я сторонник чистой командной строки, чего-то-там-box или каких-нибудь-wm – программы должны быть понятны избалованному удобствами «простому пользователю», иначе он не будет с ними работать. Но не получится ли так, что пытаясь угодить пользователям всех существующих Unix-систем, создатели настольного Unix вообще и KDE в частности на самом деле не угодят никому? LXF

Валентин Синицын » Главный редактор info@linuxformat.ru

Как с нами связаться

Письма для публикации: letters@linuxformat.ru
Подписка и предыдущие номера: subscribe@linuxformat.ru
Техническая поддержка: answers@linuxformat.ru
Проблемы с дисками: disks@linuxformat.ru
Общие вопросы: info@linuxformat.ru
Web-сайт: www.linuxformat.ru
› **Адрес редакции:** Россия, Санкт-Петербург, ул. Гончарная, 23, офис 54.
› **Телефон редакции:** (812) 717-00-37. Дополнительная информация на стр.118



Январь 2007 № 1 (87/88)

Содержание

Весь номер – прямо как на ладони: приятного чтения!

Учебники





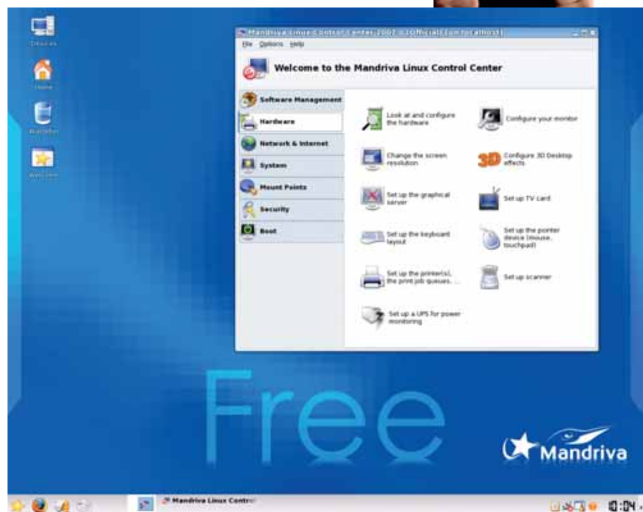
LXF DVD87/88

Mandriva Free 2007
Новая версия Mandriva включает Xgl и AIGLX, а также неплохо распознает оборудование и содержит отличные приложения. И все это доступно через фирменный инсталлятор!

Fedora Core 6
Самый быстрый релиз Fedora с крутыми 3D-эффектами. Попробуйте новый инструмент обновления, Gnome 2.16 и KDE 3.5.4!

Материалы LXF
Спецрепортажи о безопасности, настройке оборудования и ускорении системы из предыдущих выпусков журнала.

Bouncy
Выскажитесь ЗА защиту прав кроликов, установив себе эту забавную игру.



› Mandriva Linux 2007: новое ПО, новая тема, новые возможности.

Что за штука... Микроформаты?

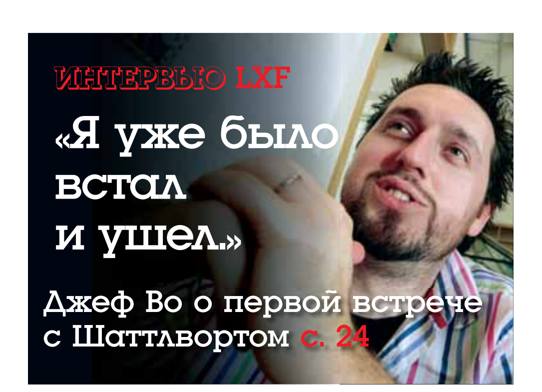
Новый смысл старых сайтов с. 36





LXFHotPicks

› Да, вы можете слушать музыку прямо в консоли!







Подпишись на Linux Format и сэкономь!



Содержание

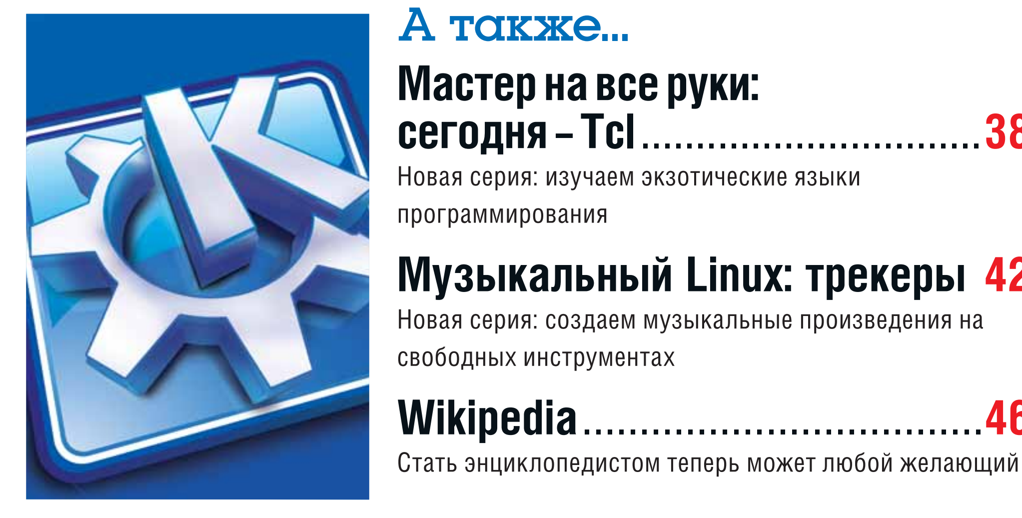
LXF DVD внутри!

См. страницу 112

Спецрепотраж

KDE 4: вкус будущего

Linux Format расследует, куда движется популярная рабочая среда с. 28



А также...

Постоянные рубрики



› Национальные дистрибутивы с.22

Обзоры



› Ubuntu: лучше чем Fedora?

Сравнение: web-браузеры



LINUX FORMAT Новости



ГЛАВНЫЕ НОВОСТИ: SCO не добилась своего » Еще одна миграция во Франции » «Бумажный» телефон под управлением Linux » Ядро 2.6.19 » ODF становится международным стандартом

» Рубрику ведет Илья Шпаньков



Linux-мобильник с бумажным экраном



Компания Motorola анонсировала выпуск давно ожидаемого мобильного телефона MOTOFONE, обладающего необычными характеристиками. Данная модель относится к супертонким – толщина корпуса составляет всего 9 мм. Другое новшество заключается в голосовых подсказках, позволяющих владельцу быстрее освоить использование различных функций телефона. При этом язык «диктора» будет соответствовать стране, в которой приобретен телефон. Но, пожалуй, наибольший интерес представляет монохромный экран устройства, выполненный из так называемой «электронной бумаги» – EPD ("electrophoretic" display), разработанной компанией E-Ink. Особенность данного экрана заключается в том, что изображение на нем по своим свойствам напоминает обычную бумагу: текст отлично читается под любым углом, не отсвечивает на солнце и практически не требует энергии для сохранения данных на экране. Последний фактор особенно полезен для значительного увеличения времени работы устройства без подзарядки – емкость аккумулятора используется только для смены изображения, после чего статичная картинка сохраняется без изменений даже при отключенном питании. По заявлениям разработчиков, заряда аккумулятора хватит на 16,5 дней в режиме ожидания и 4,5 часа в режиме разговора.

Новая модель нацелена на рынок недорогих телефонов – ориентировочная цена составляет чуть больше 50 долларов США, и по планам разработчиков подобный телефон может стать популярным среди людей пожилого возраста: этому, помимо низкой стоимости, способствует отсутствие большого набора дополнительных функций (популярных среди молодежи), а также крупный, хорошо различимый в любых условиях освещения текст на уже упомянутом выше «бумажном» дисплее. В качестве программного обеспечения в трубке используется специализированная версия операционной системы MontaVista Linux, что в большей степени продиктовано использованием необычного дисплея: именно Linux был выбран компанией E-Ink для управления «электронной бумагой» и разработки прикладных программ для устройств, ее использующих. На российский рынок данная модель должна поступить весной 2007 года.

http://direct.motorola.com/hellomoto/motofone/







Французы не любят Microsoft. Поэтому любят Linux

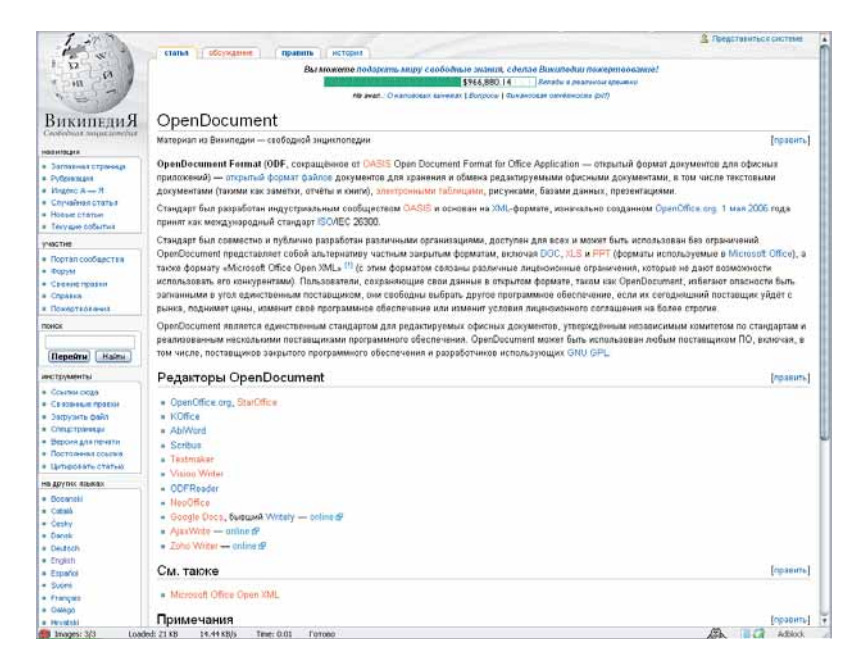


Инициативная группа, возглавляемая депутатами Ришаром Казенавом [Richard Cazenave] и Бернаром Карайоном [Bernard Carayon] предложила французским парламентариям перейти с использования программной продукции Microsoft на свободное ПО с открытым исходным кодом. По их расчетам, в этом случае затраты на внедрение и обучение персонала окажутся меньшими, чем в случае выбора в пользу программного обеспечения редмондского гиганта. По планам депутатов, более 1100 компьютеров парламентариев будут переведены под управление операционной системы Linux, в качестве офисного пакета будет использоваться *OpenOffice.org*, а *Firefox* станет основным приложением для работы в Интернете. Следует отметить, что данное предложение еще только будет рассматриваться, но многие эксперты не исключают положительного исхода. В качестве примера для подражания упоминается опыт Министерства культуры, где свободное ПО используется уже довольно активно. Также можно отметить, что Министерство сельского хозяйства Франции применяет Linux в качестве серверных систем, а французская жандармерия с успехом использует *OpenOffice.org* и *Firefox*.

Похоже, на решение французских парламентариев не окажет влияние и тот факт, что летом текущего года Министерство обороны Франции отказалось от использования свободного офисного пакета *OpenOffice.org* по причине меньшей защищенности последнего, чем *MS Office*. Правда, тестирование проводилось силами самого министерства и какого-либо заметного резонанса в обществе не вызвало.

Пока неизвестно, какой дистрибутив будет выбран для парламентариев, но, зная о «маниакальном» желании французских властей поддерживать отечественного производителя, с наибольшей долей вероятности можно предположить, что на эту роль будет выбран «родной» Mandriva (бывший Mandrake), разрабатываемый одноименной французской компанией. Тем более, что прецеденты уже были: еще в 2004 году министерство промышленности и транспорта Франции осуществило замену операционных систем Windows NT Server на MandrakeLinux Corporate Server и Windows NT/2000 Workstation на Mandrakelinux 10. В общей сложности обновление программного обеспечения коснулось 2000 серверов и около 60000 рабочих станций министерства.

OpenDocument Format признан официальным стандартом



Международная организация по стандартизации (International Organization for Standards, ISO) официально одобрила OpenDocument Format (ODF) в качестве стандарта для офисных документов. По замыслу инициаторов продвижения данного открытого формата, входящих в ODF Alliance (IBM, Sun и т.д.), он должен заменить проприетарные DOC, XLS и PPT, до настоящего времени используемые для документов, создаваемых в MS Office. Следует отметить, что компания Microsoft, в свою очередь, занимается продвижением собственного открытого формата – Microsoft Office Open XML, спецификации которого также находятся на рассмотрении в ISO. Несмотря на открытые спецификации, использование Open XML потенциально может быть затруднено в связи с различными лицензионными ограничениями. Поэтому факт признания в качестве основного стандарта ODF приветствуется большинством разработчиков офисных приложений. В частности, формат OpenDocument уже используется по умолчанию в свободном *OpenOffice.org* и коммерческом *StarOffice*, а недавно о поддержке нового формата документов объявила и компания Corel, включившая ODF в свой текстовый процессор *WordPerfect*. Впрочем, и Open XML не остается без внимания: о планах по включению поддержки данного формата в очередную версию *OpenOffice.org* объявила компания Novell.

http://www.iso.org/iso/en/commcentre/pressreleases/2006/Ref1004.html

SCO vs Linux: на «нет» и суда нет



Декабрь принес очередные новости на тему претензий компании SCO Group к разработчикам и пользователям систем GNU/Linux. Окружной судья Дейл Кимболл [Dale Kimball] после шести недель ознакомления с материалами иска, предъявленного SCO корпорации IBM, оставил в силе решение, которое ранее по этому же иску приняла судья штата Юта Брук Уэллс [Brooke Wells]. Таким образом, компания SCO Group получает отказ в удовлетворении большей части претензий к «голубому гиганту» в связи с тем, что суд посчитал недостаточными доказательства, предъявленные истцом. По словам г-на Кимболла, недовольство SCO решением суда беспочвенно, т.к. трудно было ожидать чего-либо другого, не предоставив при этом полновесных доказательств вины ответчика.

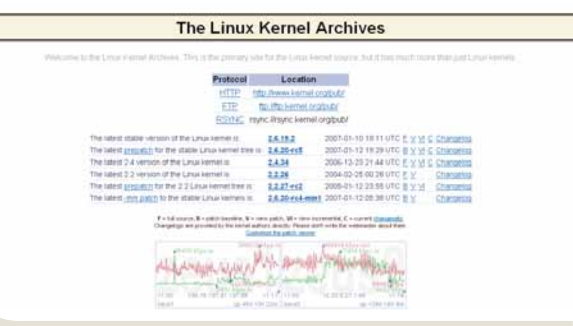
Начало разбирательств датируется 2003 годом, когда SCO обвинила компанию IBM в незаконном, по их мнению, раскрытии кода операционной системы Unix System V, авторские права на которую, опять же по словам SCO Group, принадлежат ей. При этом под «обвинения» попали и все пользователи систем GNU/Linux, построенных на ядре Linux начиная с версии 2.4 и выше, т.к. в нем специалисты SCO заподозрили наличие кода той самой Unix System V. В связи со всем вышесказанным SCO Group предложила пользователям Linux заплатить за лицензию, а со стороны IBM потребовала 5 миллиардов долларов компенсации. Между тем, за все время разбирательств каких-либо явных доказательств «вины» IBM так и не было представлено.

Оставшаяся треть претензий SCO теперь также повисла в воздухе, но уже по причине другого разбирательства – с компанией Novell, которая, узнав о претензиях к IBM, объявила в свое время, что авторские права на Unix System V принадлежат не SCO, а Novell. Именно результаты рассмотрения иска SCO к Novell теперь и станут решающими в деле SCO против IBM, т.к. в случае признания судом авторских прав Novell на злополучную UNIX-систему, все претензии к IBM вообще потеряют всякий смысл. Тот же судья Дейл Кимболл и назначил рассмотрение вопроса об авторских правах на Unix System V на 17 сентября 2007 года. Примечательно, что такой поворот событий еще больше снижает шансы SCO на положительные результаты: ранее эта компания предлагала приостановить рассмотрение иска к Novell до окончания разбирательств с IBM, что теоретически могло бы (в случае проигрыша IBM) дать более веские доказательства вины Novell. LXF

Новая версия ядра Linux: 2.6.19

Вышла новая стабильная версия ядра Linux под номером 2.6.19. В списке заметных изменений можно выделить появление поддержки кластерной файловой системы GFS2 (в добавок к уже существующей OCFS2) благодаря RedHat, выкупившей у компании Sistina права и выпустившей ее код под свободной лицензией, а также добавление в ядро поддержки ставшей вкладом компании IBM шифрованной файловой системы ECRYPTFS и экспериментальной поддержки новой версии популярной файловой системы EXT 4. Разработчики встроенных систем смогут обнаружить в новом ядре поддержку новой архитектуры AVR32, предоставленной компанией Atmel и предназначенной для 32-битных микропроцессоров RISC. Добавлено несколько новых драйверов Ethernet и SCSI-устройств, расширены опции монтирования подключаемых устройств (флэш-накопителей и медиаплейеров), улучшена поддержка устройств, подключаемых к USB-портам. Заметные изменения также коснулись работы с сетевыми протоколами и криптографических функций, встроенных в ядро.

http://www.kernel.org/



Новости короткой строкой

» Компания Novell объявила о выходе новой версии свободного дистрибутива OpenSUSE 10.2.

» Linux-пользователи, отключенные компанией Blizzard от игровых серверов World of Warcraft из-за использования Cedega, вновь допущены к игре. Они получили извинения и, в качестве бонуса – 20-дневный кредит.

» Компания RedHat объявила о выпуске новой тестовой версии Red Hat Enterprise Linux 5 Beta 2.

» Компания Nokia выбрала Red Hat Enterprise Linux для установки на все серверы предприятия.

» Компания IBM объявила о предоставлении поддержки всем, кто планирует устанавливать на серверы операционные системы Linux, в список которых входят Red Hat Enterprise Linux и Novell SUSE Linux Enterprise Server.

» Компания Grisoft подготовила бесплатную версию антивирусной программы AVG для домашнего и некоммерческого использования.

LINUX FORMAT

Обзоры



Новинки программного и аппаратного обеспечения в описании наших экспертов



Алексей Федорчук
Свою первую (и последнюю) программу написал еще на Алголе.

Быстрая загрузка – путь на рабочий стол?

Время от времени на форумах обсуждается вопрос о скорости загрузки различных ОС и дистрибутивов. В ходе этих дискуссий мне неоднократно встречалась мысль о том, что Linux (или некий его конкретный дистрибутив) грузится очень долго (по сравнению с Windows XP), и это являет собой препятствие к его распространению на рабочих столах простых пользователей.

Последнее представляется мне весьма спорным: в большинстве случаев Unix-машины используются в непрерывном или близком к тому режиме, стартуя в худшем случае раз в сутки. Однако, можно представить себе и ситуации, когда скорость загрузки оказывается важной – например, при всякого рода демонстрациях в режиме «пришел – показал – ушел». Вот я и решил проверить справедливость утверждения о медленности старта Linux-системы – в обыденной жизни я вижу его крайне редко, обычно после тотального обновления. Благо и повод подходящий представился – обновление моей Kubuntu Dapper до версии Edgy Eft, в которой впервые была применена новая система инициализации – upstart с функцией параллельного выполнения стартовых сценариев.

Измерения проводились на машине с AMD64 3500+ (реальная частота 2200 МГц). Результаты оказались следующими: примерно 32 секунды от меню GRUB до приглашения к авторизации в KDM, и не более 40 секунд – до полной загрузки KDE при автоматической регистрации в системе.

Много это или мало? Судить не берусь – тут компетентным будет мнение коммивояжера или рекламного агента на выезде. Меня – устраивает.

alv@posix.ru

Сегодня мы рассматриваем...



› Все, что можно пожелать – прямо из коробки; конечно, если все, чего вы желаете – это рюшечки.



› А вот в Ubuntu нет Xgl – не такой уж он, как оказалось, и рисковый.

НАШ ВЕРДИКТ: пояснение

Все попавшие в обзор продукты оцениваются по одиннадцатибальной шкале (10 – высшая оценка, 0 – низшая). Как правило, мы оцениваем функциональность, производительность, простоту использования и цену, а для бесплатных программ учитывается документация. Кроме того, мы всегда выставляем общую оценку, демонстрирующую наше отношение к продукту.

Выдающиеся решения могут получить престижную награду «Top Stuff». Номинантами становятся лучшие из лучших – просто высокой оценки здесь недостаточно.



Рассматривая свободное ПО, мы обычно указываем предпочтительный дистрибутив. Иногда это означает компиляцию из исходных текстов, но, если разработчики рекомендуют *Autopackage*, мы следуем этому совету.

LINUX FORMAT Вердикт

Google Earth

Разработчик: Google
Сайт: http://earth.google.com
Цена: Бесплатно по закрытой лицензии

Функциональность	10/10
Производительность	9/10
Простота использования	9/10
Документация	9/10

» Если весь мир – сцена, то Google Earth – театр. Простая в использовании, захватывающая и ободряюще практичная программа.

Рейтинг 9/10

Fedora Core 6 против Ubuntu 6.10

Громадная коммуна Ubuntu и миллионы Шаттлворта против громадной коммуны Fedora и… почем там Красная Шапка? Рефери на ринге **Энди Хадсон**.

Вкратце...

Ubuntu
» Простой в использовании дистрибутив на основе Debian. См. также Linspire, Mepis или Mandriva.

Fedora
» Передовой дистрибутив с новейшими технологиями. См. также OpenSUSE.

Нечасто мы учиняем взаимный мордобой между программами, но на этот раз решено было стравить Ubuntu 6.10 (Edgy Eft) и Fedora Core 6, и тестировать их до умопомрачения. В синем углу старая гвардия – хотя Fedora выделилась в отдельный проект всего около трех лет назад, это технологический авангард Red Hat с 1994 г. В оранжево-коричневом углу – Ubuntu, мнение о нем имеет даже тот, кто в глаза его не видал.

Впечатляющие 3.3 ГБ Fedora размещаются на 5 CD или DVD. Все, что хотите: KDE, Gnome, серверные пакеты, игры – только кликните мышкой в *Anaconda*, инсталляторе Fedora, и дело в «шляпе». К сожалению, у *Anaconda* нет средств для переразбивки винчестера, поэтому придется либо привлекать сторонние программы типа *Partition Magic*, либо ставить поверх имеющегося Linux-раздела. Не уверены, что ваш компьютер потянет Fedora? Это ваша проблема: вот поставьте, и узнаете.

Ubuntu, напротив, умещается на загружаемом Live CD – вложите его в привод, и уже ясно, сработается ли он с вашей машиной. Можно выбрать Альтернативный (скорее текстовый, чем графический) и Серверный (со скоростной технологией LAMP) режимы инсталляции. Если Gnome – не ваш выбор, придется искать другой диск (Ubuntu, Kubuntu (KDE) и Xubuntu (*Xfce*) – разные дистрибутивы).



› С помощью *Compiz* Fedora Core 6 может обрести некоторые утонченные эффекты прямо во время инсталляции.

В работе

Одно из обещаний Марка Шаттлворта сообществу Ubuntu – включить *Xgl/AIGLX* в состав Edgy Eft – похоже, не сдержано, ибо Ubuntu не поддерживает эти технологии как таковые. Их можно раздобыть через репозиторий Universe, но вы еще повозитесь, прежде чем они заработают – а ответвление *Beryl* от основной ветви *Compiz* только усугубляет проблему.

Подход Fedora более лобовой. *AIGLX* включен по умолчанию, и для его запуска надо всего лишь установить пакет *Compiz*, добавляющий новый пункт Desktop Effects в меню Preferences: откройте его, нажмите кнопку Enable Effects, и получите сворачивающиеся окна и все такое. Такая способность, да еще встроенная, действительно выводит Fedora вперед Ubuntu и поднимает планку значительно выше ваших ожиданий. На сей раз Fedora выиграла в простоте использования, что не вяжется с репутацией Ubuntu.

Канал поставок

Естественно, ни один дистрибутив не может существовать в изоляции: рано или поздно вы захотите добавить программ. А вот здесь Ubuntu напрочь прибивает Fedora. Как если бы борец вдруг схватил табуретку, чтобы напрочь вышибить дух из соперника. Происхождение Ubuntu от Debian дает ему громадное преимущество благодаря пакетному менеджменту *apt-get*. Fedora же полагает-

Управление пакетами в сравнении



Apt-get
Ubuntu может сортировать пакеты по степени популярности – такую бы легкость да любому пакетному менеджеру.



Pirut
Pirut, напротив, просто портит в остальном превосходный дистрибутив.

На чашах весов

	Ubuntu	Fedora
Время загрузки*	41 сек.	76 сек.
Размер дистрибутива	One CD	Five CDs
Доступно пакетов	16,000	6,000
Время инсталляции	15 мин.	25 мин.

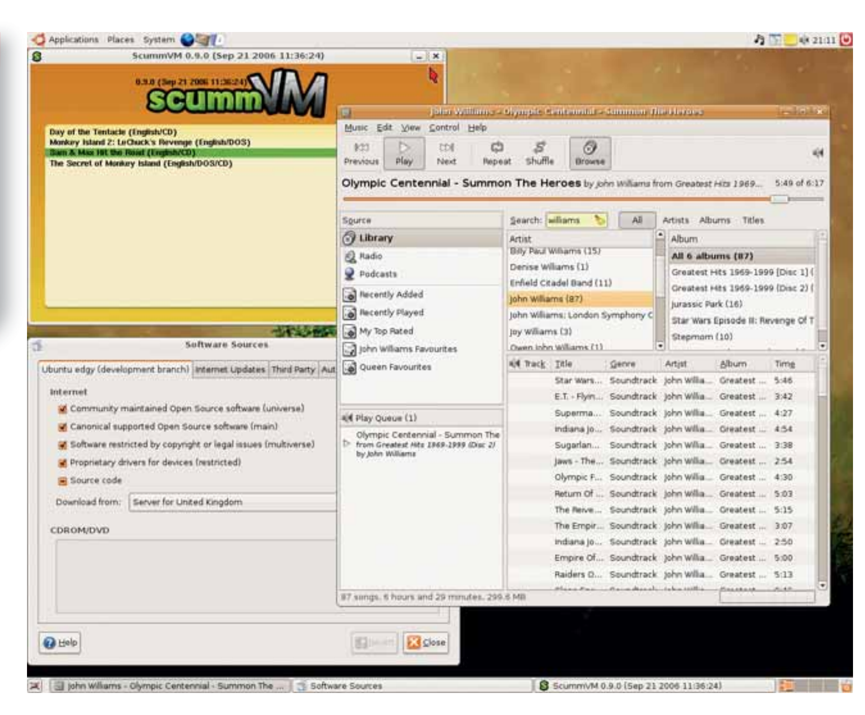
*Pentium 4, 3.4 ГГц, 1 ГБ ОЗУ, видеокарта ATi 9600 XT, жесткий диск 250 ГБ SATA.

ся в основном на *Yum* с его армией RPM. При всем уважении к разработчикам *Yum*, *apt-get* неизмеримо совершеннее и быстрее. Будь то командная строка с **apt-get update** и **apt-get dist-upgrade**, *Update Manager* или *Synaptic* – *apt-get* мгновенно откликается и предоставляет пользователю мощный и простой механизм установки новых пакетов. Заметная работа была проделана в настройке репозиториев Ubuntu, включая совершенно новый графический инструмент *Software Sources*. Он не только облегчает работу с файлом **sources.list**, но и управляет импортом GPG-ключей. Высший балл Ubuntu за такое удобство прямо «из коробки».

> «Желающим попробовать Xen без Fedora не обойтись.»

Грубо контрастирует со всем этим *Pirut*, замена набившего оскомину *Add/Remove packages*. Впервые *Pirut* увидел свет в FC5, затем с малыми поправками дошел до FC6, но ничуть не повзрослел. Основная его ошибка – надежда на то, что всю черновую работу в фоновом режиме сделает *Yum*. При своей некосмической скорости, *Yum* только тянет назад любое основанное на нем приложение, поэтому *Pirut* ужасно медлителен. Еще один досадный минус – сложность доступа к RPM на CD или DVD. Если не оговорить явно, что нужные пакеты находятся в основной инсталляции, вас пошлют за ними подальше – аж в Интернет. А как же громадный склад программного обеспечения на локальной оптике? Недоразумение какое-то. Механизм установки софта в Ubuntu – сама простота рядом с его подобием в Fedora.

Зато реальная победа Fedora – интеграция *Xen*. Уничижительные отзывы Red Hat о, пожалуй, преждевременном включении *Xen* в решения Novell оказались слишком верны. В Fedora подождали, пока технология созреет, представив прекрасный гипервайзер под названием *Virtual Machine Manager* и ряд других усовершенствований. Теперь желающим попробовать *Xen* без Fedora не обойтись. У нас просто слюнки текут в предвкушении Red Hat Enterprise Linux 5 с поддержкой *Xen*.



› Благодаря Gnome 2.16 с Cairo 1.2, и Fedora Core 6, и Ubuntu Edgy Eft (это он) имеют приятные виджеты векторного производства и четкие, чистые линии.

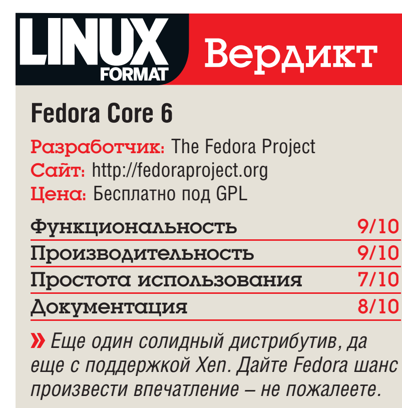
Мелкие штрихи

В этой области Ubuntu похвастаться почти нечем. Да, можно установить *Xen*, да, есть некие утилиты для управления им, но с лоском и легкостью Fedora – никакого сравнения. Вероятно, Ubuntu сдал позиции за счет четырехмесячного цикла разработки против шести месяцев у Fedora, из-за переключения работ на Dapper (итог которых – отличный дистрибутив), плюс упорства, с которым Марк Шаттлворт придерживается октябрьско-апрельского расписания.

Из слова Edgy («рисковый») в названии следует, что Шаттлворт не рассчитывал на особую стабильность дистрибутива, но мы почти не ощутили никаких проблем. Выглядит, как слегка продвинутый Dapper, только без трехлетней поддержки. Есть надежда, что Edgy +1 шагнет дальше в технологичности и функциональности, но сейчас особого искушения перейти с Dapper на Edgy нет. Обычно мы не скупимся на похвалы Ubuntu, но на сей раз будем сдержанны. Так неужели это бросовый продукт? Отнюдь нет: релиз выдающийся, как и Dapper Drake. Может, он и поотстал от Fedora, но лишь из-за разницы в возрасте.

А вот пользователям Fedora следует обновляться до FC6 немедленно. Fedora Project выдал дистрибутив, достойный внимания. LXF

LINUX FORMAT Вердикт



Fedora Core 6

Разработчик: The Fedora Project
Сайт: http://fedoraproject.org
Цена: Бесплатно под GPL

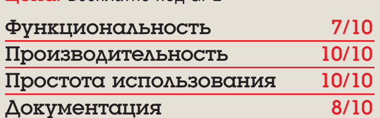
Функциональность	9/10
Производительность	9/10
Простота использования	7/10
Документация	8/10

» *Еще один солидный дистрибутив, да еще с поддержкой Xen. Дайте Fedora шанс произвести впечатление – не пожалеете.*

Рейтинг 8/10

Ubuntu 6.10

Разработчик: Canonical
Сайт: www.ubuntu.com
Цена: Бесплатно под GPL



Функциональность	7/10
Производительность	10/10
Простота использования	10/10
Документация	8/10

» *Перейти от Dapper к Edgy? Скорее, переползти. Обновляйтесь в случае крайней необходимости.*

Рейтинг 9/10

LINUX FORMAT TOP STUFF

FreeBSD 6.2

Основа крупнейших web-сайтов мира, FreeBSD прячет свою надежность и легкий вес под спудом тяжких серверов. **Майк Сондерс** расколупывает новую версию.

Вкратце...

» Unix-подобная система для рабочих станций и серверов для опытных пользователей. См. также: NetBSD, OpenBSD и, конечно же, Linux!

Вопрос: назовите открытую, разработанную сообща, Unix-подобную операционную систему, основу мегапортала Yahoo! Многим тут же придет на ум Linux, но если в Linux вы уже не новичок, то, несомненно, встречали в Сети упоминания о FreeBSD (увы, большей частью в перепалках типа «а моя ОС лучше вашей»). А тот, кто полистал LXF77, наверняка заметил статью о FreeBSD и понял, что она заслуживает большего, чем позволяют наши колонки. Простите нам секундное отступление, пользователи FreeBSD: мы уже переходим к разбору 6.2.

FreeBSD происходит от операционной системы Berkeley Software Distribution, разработанного в начале девяностых варианта Unix для универсальной платформы x86 PC. Мгновенно завоевав успех у провайдеров и администраторов серверов, FreeBSD тут же увязла в юридических тяжбах с владельцем Unix, AT&T, предоставив Linux оттянуть на себя все внимание публики.

К счастью, FreeBSD выстояла в юридических битвах, выиграла лицензионные войны и теперь играет главные роли в мире открытого ПО (вы уже догадались: за спиной Yahoo.com стоит именно она). Во многих отношениях FreeBSD подобна Linux: это система с открытым кодом, похожая на Unix, предназначенная для настольных компьютеров и серверов, с набором хорошо известного ПО, включая KDE, Gnome, *Firefox* и 95% других программ, используемых под Linux. Устанавливается она без услужливого «мастера», но если вы дружите с командной строкой, проблем с FreeBSD у вас не будет, учитывая стройную систему директорий и превосходную документацию.



» FreeBSD отлично ладит с Linux-программами, включая *Firefox* и *IceWM* (на рис.).

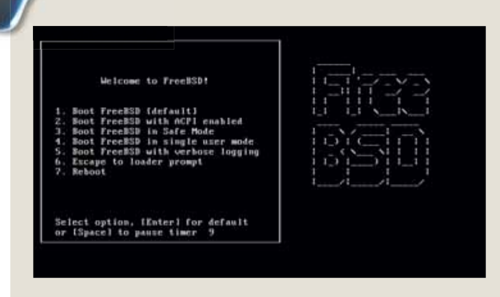
> «Через систему Ports доступно свыше 16 000 программ.»

Кое-что старое

Итак: загрузчик FreeBSD 6.2, *Sysinstall*, остался тот же, что и в предыдущих версиях 6.x, он шустро проходит процесс установки с текстовыми меню и диалоговыми окнами, позволяя вам изменить стандартный набор ПО и настройки сети, хотя здесь уже заметно отставание от дистрибутивов Linux по части разбивки диска – например, нет способа потеснить существующие Windows-разделы. Серверу это ни к чему, но пользователям настольных компьютеров и рабочих станций очень бы пригодилось.

Установленная FreeBSD 6.2 внешне почти неотличима от 6.1: очень быстрый старт (до появления текстового приглашения login 26

Шаг за шагом: Установка FreeBSD



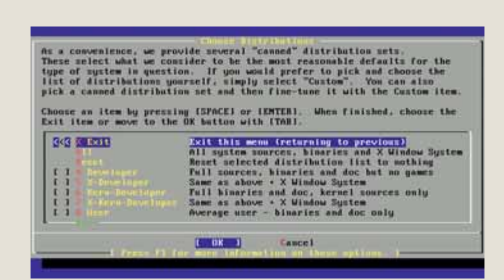
» Загрузка

Загрузитесь с CD и при виде загрузочного экрана нажмите Enter. (Выберите вариант Safe, если случатся проблемы.) Укажите язык и клавиатуру, затем выберите Стандартную установку.



» Разбивка жесткого диска

FreeBSD нарезает диск на «слайсы» (аналог разделов), в которых создаются секции FreeBSD (подразделы). Как минимум, необходимы корневой раздел (/) и раздел подкачки (swap).



» Свободное ПО для всех

Уточните свои предпочтения (большинство устроит набор «X-User»), и инсталлятор скопирует необходимые файлы. Здесь показан вариант финальной настройки перед перезагрузкой во FreeBSD.

секунд на нашем 2-ГГц тестовом компьютере, против 47 секунд Ubuntu), простая оболочка (*Bash* тоже имеется) и голая система, на которую вам предстоит навесить любимые приложения. Документация по-прежнему первоклассная, есть детальные руководства по любому аспекту системы (включая драйверы) и превосходный учебник, в котором подробно описаны мельчайшие винтики FreeBSD.

Свыше 16 000 программ доступны через систему Ports («порты») в виде исходников, многое можно загрузить как двоичные пакеты. Для запуска сугубо Linux'овых приложений в FreeBSD предусмотрен слой совместимости, где могут работать *Oracle*, *SAP* и *Matlab*. Если взять библиотеки и инструменты пользовательского пространства Linux (то есть зоны взаимодействия пользователя с ОС, будь то рабочий стол или командная строка) и чуть-чуть доработать системные вызовы, то многие Linux-программы почувствуют себя, как дома.



› *Sysinstall*, загрузчик FreeBSD, исправно работает, не меняясь годами – но ему не хватает средства разбивки жесткого диска.

Кое-что новое

Самое заметное для пользователей изменение в 6.2 – это включение *FreeBSD Update*. Это крупный шаг вперед: теперь пользователи могут загружать обновления безопасности, вместо того чтобы выискивать изменения на *CVS* и всякий раз пересобирать ядро, пользовательские программы и библиотеки. Установка полного комплекта инструментов разработчика на действующий сервер – серьезный риск, поэтому здорово, что FreeBSD справилась и с этим, подойдя еще ближе к Debian и Co.

Аудит безопасности теперь включен в ядро, что значительно облегчает слежение за критическими в этом отношении системными событиями, типа подключений, изменений в файлах конфигурации и выходов на сеть. В сопровождении пользовательских утилит и файлов конфигурации, новая система безопасности была импортирована из проекта OpenBSM, и пока имеет статус экспериментальной. Впрочем, при послужном списке, где значатся McAfee, Apple и TrustedBSD, OpenBSM выглядит достаточно солидно, чтобы стать постоянной частью ОС.



Пол считает...

«FreeBSD была моим первым Unix-ом, и она до сих пор хороша для серверов. Но я бы оспорил включение *MySQL 5.1* – все же это только бета!»

Большая часть сторонних программ обновлена, включая *GCC 3.4.6*, *OpenSSH 4.4p1* и *Sendmail 8.13.8*, значительная работа проделана в системе Ports, и теперь доступны KDE 3.5.4, *Gnome 2.16.1*, *Firefox 2.0* и *OpenOffice.org 2.0.4*. Администраторы серверов найдут, что *Apache* обновлен до 2.2.3, а *MySQL* – до 5.1.11. На аппаратном фронте разработчики получили FreeBSD, работающую на консоли Xbox, а для удаленного управления и контроля добавлен драйвер IPMI.

Linux-совместимость усовершенствована добавкой драйвера *linsysfs*. Это дает FreeBSD собственный вариант директории **/sys**, где собраны сведения о конфигурации компьютера. Это еще не способ заставить работать Linux-приложения напрямую, но подготовка почвы для должного поведения программ, рассчитанных на конкретное оборудование.

Кое-что грустное

Сторонники FreeBSD, понятное дело, заявляют, что система не конкурирует с Linux, она может выжить сама по себе. Коллективный разум, однако, очень важен: если FreeBSD будет цепляться за пережитки Unix, ей не видать разработчиков и тестеров в количестве, необходимом для развития. Конечно, FreeBSD не нуждается в многомиллионных финансовых вливаниях, но если ее оттеснят на обочину крупные Linux-брэнды, новых разработчиков будет найти трудно. Следовательно, сообществу FreeBSD необходимы компромиссы. Идея творящих чудеса мастеров установки абсолютно чужда любому Unix-хакеру, но почему бы не добавить пару инструментов настройки, хотя бы систему настройки X для рабочих станций? Или несколько диалоговых окон для настройки комплекса FAMP (FreeBSD, *Apache*, *MySQL*, Perl), чтобы вечно занятые администраторы могли развернуть FreeBSD за несколько минут?

Матерого пользователя FreeBSD такие советы наверняка покоробят – с него довольно приглашения оболочки. И это здорово. Но ведь инструменты настройки, удачно работающие с текстовыми файлами конфигурации, вовсе не утопия, что доказано Red Hat. Если команда FreeBSD соединит гениальную простоту системы с новейшими инструментами настройки, все от этого только выиграют. Это будет полезно и для настольной FreeBSD, и для сервера.

А еще кое-что?

Мы говорим не о привлечении масс, а об облегчении жизни администраторов, которые приветствуют идею FreeBSD, но нуждаются в некотором ускорении работы. FreeBSD по-прежнему в основном серверная система, и из-за сравнительно короткого цикла поддержки (два года против пяти у Ubuntu) и злоупотребления клавиатурой многие обходят ее стороной.

Отложив все это в сторону, скажем, что FreeBSD – превосходная ОС, более цельная и понятная в повседневной работе, чем продукция рассыпной методологии Linux. Все в ней плотно подогнано, а не слеплено как попало – заметна забота и внимание к деталям. Если команда FreeBSD сумеет использовать потенциал системы, дополнив его новейшими инструментами в помощь администратору, ОС ждет завидное будущее, а Linux получит сильного конкурента. FreeBSD 6.2 – эволюционная версия сверхстабильной серии, и в ней достаточно лакомых кусочков, чтобы захотелось до нее обновиться. LXF

Ключевое ПО FreeBSD

	FreeBSD 6.1	FreeBSD 6.2
X.org	6.9.0	6.9.0
KDE	3.5.1	3.5.4
Gnome	2.12.3	2.16.1
Firefox	1.5.0.1	2.0
OpenOffice.org	1.1.5	2.0.4
Apache	2.2.0	2.2.3
MySQL	5.1.6	5.1.11
PostgreSQL	8.1.3	8.1.4
PHP	5.1.2	5.1.6
Sendmail	8.13.6	8.13.8
Postfix	2.2.9	2.3.3

LINUX FORMAT Вердикт

FreeBSD 6.2

Разработчик: The FreeBSD Foundation
Сайт: www.freebsd.org
Цена: Бесплатно под лицензией BSD

Функциональность	7/10
Производительность	9/10
Простота использования	5/10
Документация	10/10

» Типичный набор FreeBSD: надежность, быстрота и четкость дизайна; но – как бы не подрезали серверные дистрибутивы Linux!

Рейтинг 8/10

Mandriva Powerpack 2007



Реклама преподносит новейшую версию этого дистрибутива как квантовый скачок. В свете ухода Гаэля Дюваля, **Грэм Моррисон** интересуется: а куда скакнули-то?

Вкратце...

» Linux становится ближе. Mandriva легко инсталлировать, им приятно пользоваться, имеется достойная библиотека приложений. См. также: Mepis или Ubuntu.

Mandriva никогда не гналась за переменами ради перемен. Процедура ее установки за пять лет почти не изменилась. Это совсем неплохо: превосходный инсталляционный процесс и непривередливость к оборудованию всегда были двумя основными причинами выбора Mandriva и его популярности в качестве первого дистрибутива для новичков в чудесном мире Linux. Хронически меняется только логотип.

Mandriva Linux 2007 – девятнадцатая версия со времени образования дистрибутива в 1998, и первая после ухода из проекта его основателя, Гаэля Дюваля. Рассматриваемый здесь Powerpack – один из пяти отдельных, хотя и взаимосвязанных, дистрибутивов семейства 2007. Основной дистрибутив – Free, бесплатный, в духе этики прародителя Mandrake. Затем идет Mandriva One, общедоступный инсталлируемый LiveCD, включающий коммерческие драйверы. Есть также три платные коробочные версии: Discovery, Powerpack и Powerpack+. Каждая из них, наряду с платными Linux-приложениями, включает в себя различное количество свободного ПО, документацию и поддержку.

Инсталляция Powerpack продолжалась 45 минут, в основном из-за копирования всего 4,7-гигабайтного DVD на жесткий диск, хотя и необязательного (без полного копирования установка проходит гораздо быстрее), но стоящего затраченного времени: не придется потом откапывать DVD в поисках дополнительных пакетов. Мы были растроганы тем, что Mandriva распознал наш 1400x900 широкоэкранный TFT-монитор и даже спросил, не желаем ли мы установить коммерческий драйвер ATI для нашей видеокарты.



› Выглядит Mandriva привлекательно, *RealPlayer* запускается автоматически. Только он почему-то не встроен ни в *Firefox*, ни в *Konqueror*.

Покинуты во мраке

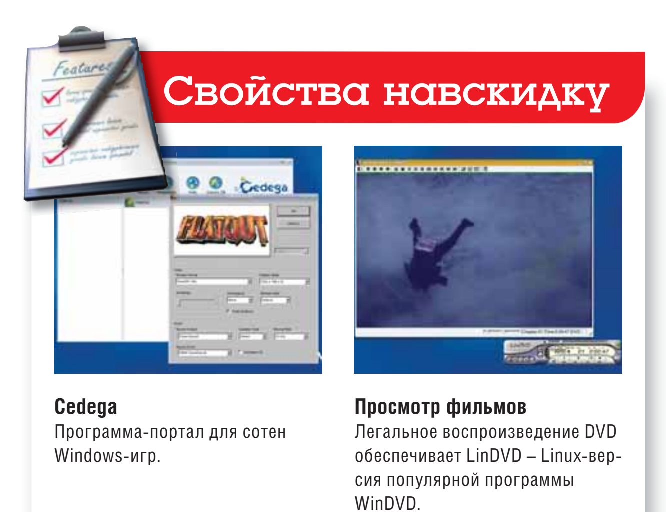
Счастье было недолгим. По завершении установки нас встретил совершенно пустой экран. Настройка X была неправильной, положение не спас даже выбор стандартной настройки инструментом *DrakX11*. Пришлось вручную редактировать **xorg.conf** и убирать оттуда в корне ошибочные частоты обновления экрана, прежде чем мы смогли что-либо увидеть на мониторе. Первоначальные параметры могли не только повредить аппаратуру, но и оставить Linux-новичка буквально во мраке.

Приятным моментом можно считать легкость настройки нашей восьмикнопочной мыши: каждая из кнопок получила четкие инструкции как для вертикальной и горизонтальной навигации по web-страницам, так и для перемещения вперед/назад. Рабочий стол по умолчанию по-прежнему KDE, но Gnome можно выбрать при установке одним щелчком.

Система грузится сравнительно быстро: примерно 30 секунд от BIOS до рабочего стола – почти вдвое быстрее Ubuntu Dapper. Ну, а когда загрузился рабочий стол, то и жаловаться на отсутствие стоящей темы не пришлось. Экран обильно орошен фирменным Royal Blue, да еще заметно улучшенным по сравнению с 2006, главным образом благодаря новому стилю под названием la Ora (видимо, «привет» на французско-полинезийском языке). Успех la Ora состоит в отказе от надоевших раскрашенных скосов и контрастных границ. В результате получился мягкий спокойный дизайн, напоминающий стандартную тему Ubuntu. Кстати, выбор цветовой схемы включает в себя коричневую палитру Ubuntu, но замена нежно-голубых оттенков на кислотно-оранжевый может выглядеть прилично только на мониторе, поврежденном во время процесса инсталляции.

А вот и изюминка. Дополнительная панель управления, под названием 3D Desktop Effects, подарит вам вращающийся куб, без компиляции и поиска зависимостей. Панель подключает «Все объемные эффекты» с помощью чудесного *Compiz* и *Xgl* или *AIGLX*. Предпочтение отдается *AIGLX*, и работает все прекрасно. Даже с KDE, прозрачные границы окон и полутени смотрятся замечательно.

Свойства навскидку



Cedega
Программа-портал для сотен Windows-игр.

Просмотр фильмов
Легальное воспроизведение DVD обеспечивает LinDVD – Linux-версия популярной программы WinDVD.

Ключевое ПО Mandriva

Свободное:
- Ядро 2.6.17
- Gnome 2.16
- KDE 3.5.4
- *GCC 4.1*
- *glibc 2.4*
- *Mozilla 1.5.0.6*
- *OpenOffice.org 2.0.3*
- *X.org 7.1*
- *Xen 3.0*

Коммерческое:
- *Adobe Reader*
- *BitDefender AntiVirus*
- *Cedega*
- *Helix RealPlayer*
- *Kaspersky Anti-Virus* (включая 6 месяцев обновлений)
- *LinDVD*
- *VMware Player*

› Включение коммерческих драйверов означает, что вы получаете превосходную производительность 3D, без которой не обойтись в игре *CanonSmash*.

Mandriva использует для настройки инструмент *gset-Compiz*, его стандартные параметры вполне разумны (в смысле, не злоупотребляют раскачкой окон до степени морской болезни). Несмотря на это, *Compiz* пока еще носит статус экспериментального. После нескольких часов его использования производительность нашей системы значительно снизилась. Но с этим мы сталкивались и в других дистрибутивах, поэтому ошибки скорее не в Mandriva, а в самой технологии.

Еще одно крупное отличие этой (и остальных платных) версии – включение коммерческого программного обеспечения. Верхнюю строчку в списке «переключись на Linux» занимает *Cedega* от Transgaming, умеющая запускать многие Windows-игры. Не нужно устанавливать ее дополнительно, она включена по умолчанию – просто щелкните по значку в меню Игры. В доказательство даже прилагается Windows-игра образца 2004 г. под названием *FlatOut*. Она не шибко оригинальна – напоминает *Destruction Derby* – но физика и графика неплохо смотрятся на бедной текстурами Linux-платформе. *Cedega*, как обычно, требует как минимум 2 ГГц машины, а видеокарту лучше иметь Nvidia с коммерческими драйверами – ускоритель ATI испытывал трудности даже с предустановленной *FlatOut*. Но если машина точно соответствует спецификации, можно рассчитывать на игры-гиганты типа *Half-Life 2*, *Battlefield 2* и *World of Warcraft*. Включение *Cedega* в Mandriva означает, что вы не получите обновлений (обычно этот сервис стоит $15 в квартал), зато вам станут доступны сотни игр.



Пол считает…

«Нелегко критиковать превосходный дистрибутив — я полагаю, что Mandriva опередил и Fedora, и SUSE, но вот до Ubuntu чуть-чуть не дорос.»

Новости мультимедиа

Интересно было взглянуть на *LinDVD* от Intervideo – специальный Linux-порт популярной программы *WinDVD*. Linux-программе до *WinDVD* далеко: интерфейс основан на продукте четырехлетней давности, есть, похоже, и проблемы со стабильностью; но со всем этим можно примириться ради готового и легального воспроизведения DVD. К тому же DVD – не единственный коммерческий формат, доступный в Mandriva. Для воспроизведения аудио/видео Real включен Helix *RealPlayer* (правда, почему-то не встроен ни в один из браузеров), а инсталляция *Skype* доступна в один щелчок. Пользователям Windows все это существенно, поэтому Mandriva – неплохой выбор для мигрантов.

По примеру Ubuntu, уменьшено количество значков на рабочем столе – осталась лишь корзина. В KDE легко растеряться, не зная, где найти простейшие вещи, вроде файлов в домашней директории, особенно не будучи в курсе, что *Konqueror* – это еще и файловый менеджер: ведь здесь, в отличие от Ubuntu, значка *Konqueror* нет в панели инструментов. Пользователям Gnome будет полегче: у них есть меню Places. Еще одна проблема – подключенный USB-накопитель не появляется на рабочем столе. KDE автоматически открывает файловый браузер, но нет способа вручную отключить устройство, чтобы убедиться в сохранности изменений. Выход есть, конечно – можно набрать в адресной строке *Konqueror* **system:/** – но ведь не всякий догадается! А вот найти приложение через меню теперь стало как никогда просто. Связано это с отказом от системы меню Debian, используемой со времен Mandrake 7.1. Новая система называется XDG и является стандартом и для KDE, и для Gnome.

Так куда движется проект? В целом, эта версия мощнее последних двух. Есть проблемы по совместимости с оборудованием, но они и всегда были. Стабильность дистрибутива делает его реальным кандидатом на звание «Мой первый Linux». Если вы преданный пользователь Mandriva, эта версия вас приятно удивит, а если новичок – то это хороший выбор по мультимедиа, Windows-играм и воспроизведению DVD. При этом Mandriva включает в свои коробочные версии внушительное собрание коммерческих программ. А вот те, кому нужна вся мощь Linux, переходить не поспешат. LXF



› В Mandriva проделали большую работу по интеграции *Compiz* – но только ли нам надоели эти кубические снимки экранов?



LINUX FORMAT Вердикт

Mandriva Powerpack 2007

Разработчик: Mandriva
Сайт: www.mandriva.com
Цена: 1250 руб. [для России]

Функциональность	7/10
Производительность	8/10
Простота использования	8/10
Документация	7/10

» *Впечатляет: быстрый дистрибутив с некоторыми новейшими функциями и богатым набором коммерческого ПО.*

Рейтинг 8/10



Oxygen 7.2

Создавать структурированный текст надо играючи! Встречаются два редактора: **Ник Вейч** и достойный редактор XML...

Вкратце...

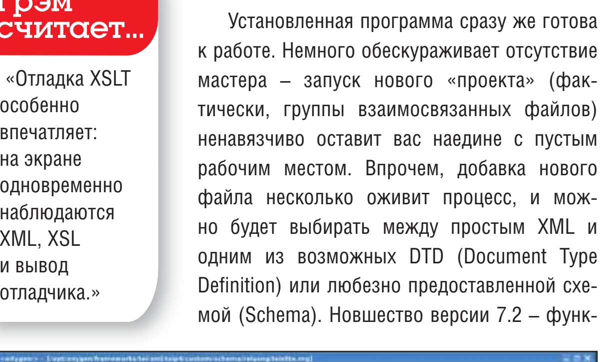
» XML-редактор. См. также стандартные редакторы с XML-функциями, такие как *Kate*, *Emacs* или *Vi*.

Разве обязательно писать XML специальным инструментом? В общем, нет – но ведь и стиральную машину заводить необязательно. Можно все это делать вручную, да только придется отвлечь часть интеллекта на то, что вы в шутку называете сантехникой. *Oxygen* – не столько редактор, сколько полноценная среда разработки: фабрика по выработке, отладке и тестированию XML-файлов. Фактически, это даже нечто большее, так как здесь есть средства и для создания XSLT, и для отладки XQueries.

> «Совместный доступ к файлам через FTP или Subversion – легко.»

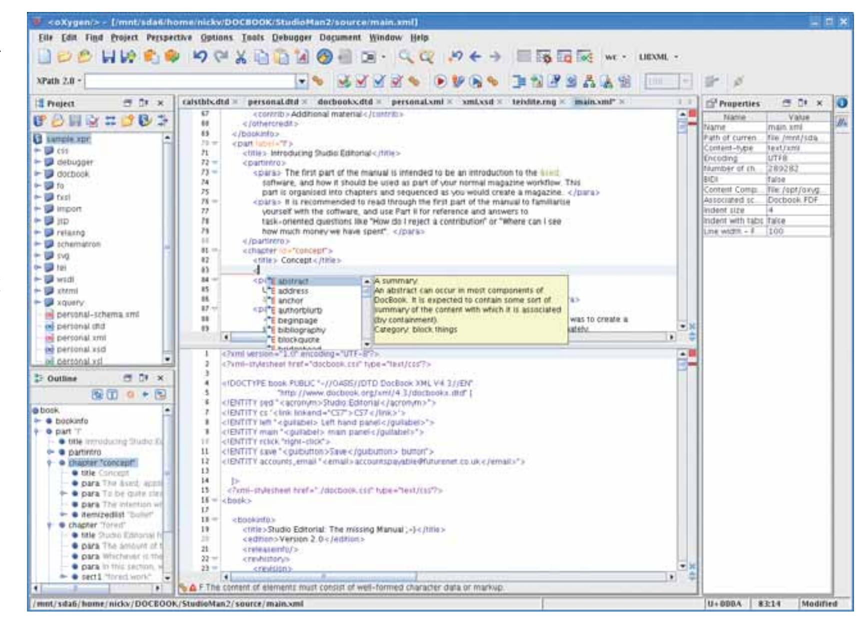
Oxygen – Java-приложение. Вы издали стон? Сдержитесь: все выглядит достойно и работает быстро. Правда, бывают проблемы с разными реализациями JVM. *Oxygen* тестировался со стандартной версией Java от Sun, а с реализацией Java от *GCC* работает плохо, поэтому если у вас Fedora или Debian, придется переступить через себя и установить «нечистое».

Установленная программа сразу же готова к работе. Немного обескураживает отсутствие мастера – запуск нового «проекта» (фактически, группы взаимосвязанных файлов) ненавязчиво оставит вас наедине с пустым рабочим местом. Впрочем, добавка нового файла несколько оживит процесс, и можно будет выбирать между простым XML и одним из возможных DTD (Document Type Definition) или любезно предоставленной схемой (Schema). Новшество версии 7.2 – функция изучения DTD по созданному вами XML. Структура, которую вы получите, будет чем-то вроде наименьшего общего знаменателя, но она может лечь в основу более сложному определению.

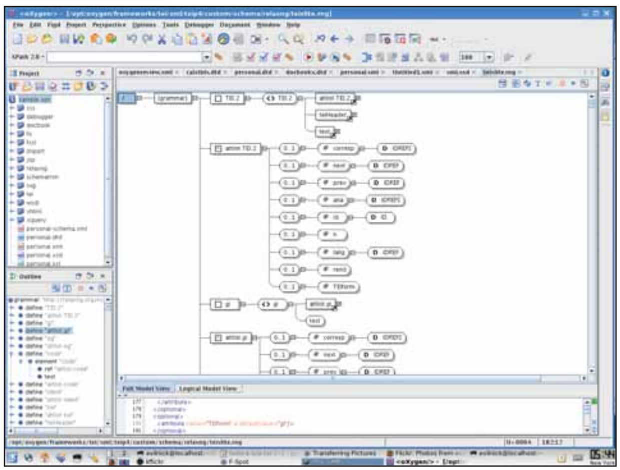


Грэм считает...

«Отладка XSLT особенно впечатляет: на экране одновременно наблюдаются XML, XSL и вывод отладчика.»



» Это не просто редактор – он больше похож на XML-дружка; с ним весело!



» Визуальное редактирование файлов RelaxNG – не просто трюк.

Роскошь стала щедрее

Конструирование и обработка схем может таить неприятности, так как структуру очень легко изменить, самому того не заметив. Вот почему один из самых полезных инструментов *Oxygen* – экран визуального моделирования, наглядно представляющий устройство структуры. При желании его можно экспортировать как графику или распечатать в размере плаката.

При групповой или удаленной работе можно легко организовать обмен файлами через FTP или *Subversion*. Клиент *Subversion* принимает вид дополнительного диалогового окна, а для выполнения операций *add* и *commit* можно пользоваться контекстным меню. Для недоверчивых есть методы включения цифровой подписи, удостоверяющей владельца, но поддерживаются пока лишь схемы JKS и PCKS12.

Oxygen использует ряд стандартных внешних инструментов, например, различные версии *Saxon* или FOP от *Apache*; последний производит HTML из XML. Единственная проблема – *Apache* FOP спотыкается на некоторых сложных структурах (в частности, DocBook), но его нетрудно заменить коммерческим FOP. Если зашли в тупик – к вашим услугам превосходная документация. Цена [для Великобритании, – прим. ред.] (а это коммерческое ПО), как говорится, дешевле краденого.

На нынешней стадии развития *Oxygen* выше всяких похвал. Все обычные полезности (рефакторинг, умное завершение, diff/merge, подсветка синтаксиса...) улучшены и расширены до такой степени, что любое новшество воспринимается уже как шоколадная крошка поверх аппетитной булочки. Создание XML другим способом трудно даже представить. LXF



LINUX FORMAT Вердикт

Oxygen 7.2

Разработчик: SyncRO Soft Ltd

Сайт: www.oxygenxml.com

Цена: Учебный/домашний вариант $48 без НДС, профессиональная версия $225 без НДС

Функциональность	10/10
Производительность	9/10
Простота использования	9/10
Цена	8/10

» *Всем бы редакторам такую мощь и гибкость!*

Рейтинг 9/10

Valgrind 3.2.1



Грэм Моррисон спустился в недра отладки, а путь ему освещал *Valgrind*, всевидящий ловец ошибок.

Вкратце...

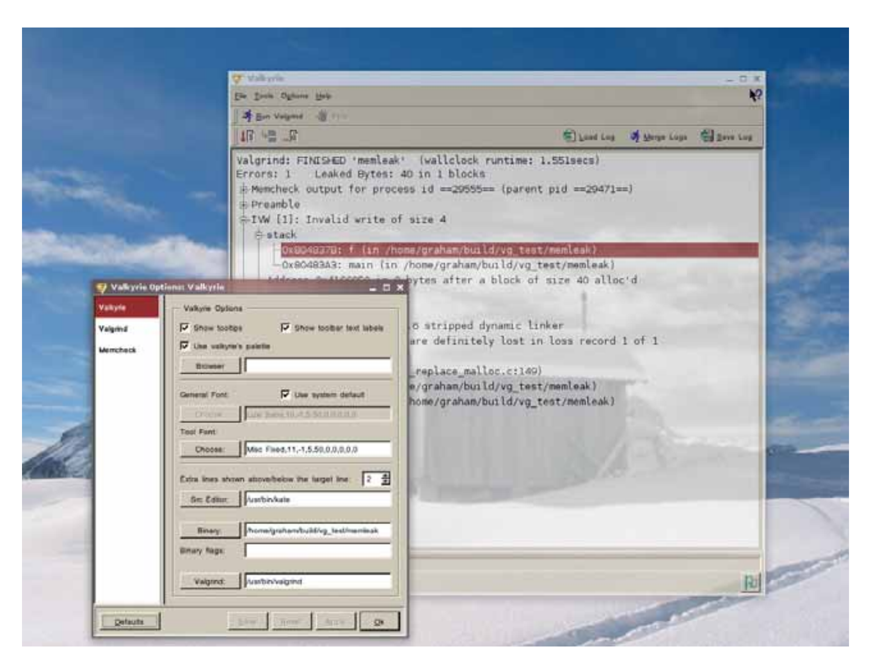
» Запускает вашу программу в виртуальной машине, которая отлавливает ошибки работы с памятью и может составить профиль производительности. См. также: *UndoDB*.

За четыре года, прошедшие со дня его выхода в свет, *Valgrind* стал неотъемлемой частью процесса программирования на C/C++. Этот вклад был отмечен премией Google–O'Reilly Open Source Award на Open Source Conference (OSCon) 2006 года, а *Firefox*, *OpenOffice.org*, *Opera*, KDE, Gnome и Unreal Tournament имели бы куда больше проколов, если бы не *Valgrind*. И уж конечно, все они были бы медлительнее – кроме, пожалуй, *OpenOffice.org* (куда уж еще-то?).

У *Valgrind* нетипичный подход к отладке: он встраивает вашу программу в собственную виртуальную машину и может отследить любую команду и выделение памяти, контролируя каждый регистр и байт. В результате возникают две проблемы. Во-первых, вы привязаны к интерпретации процессора *Valgrind*'ом, и если ваша программа полагается на специфическую команду процессора, *Valgrind* обязан ее поддерживать. К счастью, отладчик постоянно совершенствует свой арсенал, и к новейшей версии *Valgrind* добавлены 64-битные команды AMD и поддержка почти всех команд SSE3, имеющихся в новейших Pentium 4 и Athlon. Трудности могут быть только у тех, кто компилирует программное обеспечение на ультрасовременных процессорах.

> «Вам понадобится мощная машина и терпение святого.»

Вторая проблема – *Valgrind* тормозит. С учетом объема работы по слежению за каждым действием вашей программы, это неизбежно, и все же неповоротливость отладчика шокирует. Старт *Konqueror* в KDE на нашем компьютере с CPU 3 ГГц занял 1,5 минуты (обычно – 3 секунды), а общение с браузером замедлилось раз в 20. Но при этом было отмечено увеличение скорости на 15 % по сравнению с прежней версией *Valgrind* (3.1.0), так что положение улучшается.



› По умолчанию результат выводится в файл. GUI вроде *Valkyrie* очень облегчает работу.

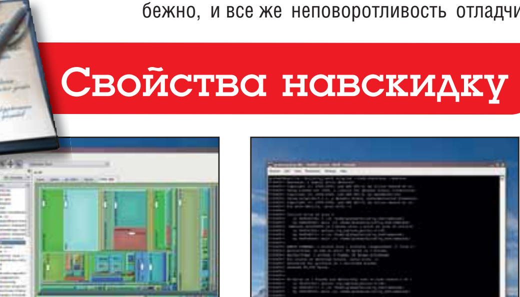
Valgrind не просто указывает на ошибки в вашем собственном ходе, он отмечает таковые во всех библиотеках, которыми вы пользуетесь. Этот список может стать чудовищным: за время типового сеанса генерируются сотни ошибок – даже если ваш код безупречен. Хорошо еще, что вывод можно фильтровать, используя заготовленный профиль. Например, профиль KDE обходит все ошибки, найденные в библиотеках KDE, облегчая вам поиск собственных промашек. *Valgrind* может форматировать список в виде XML, и с ним можно работать в графической оболочке при помощи отдельной программы *Valkyrie* или среды разработки *KDevelop*.

Грызя гранит

Метод виртуальной машины, применяемый *Valgrind* для контроля памяти, оказался настолько плодотворным, что некоторые другие пакеты инструментов прибрали его к рукам. Среди них *Massif* для проверки использования памяти; *Helgrind* для отлова состояний гонки (race) в многопоточном коде; и *Cachegrind*, отмечающий время, затраченное процессором на каждую из функций.

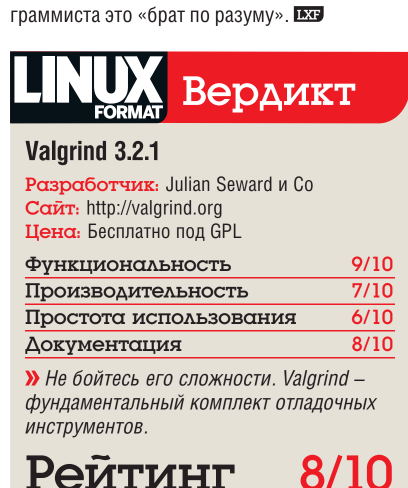
Процесс невероятно ресурсоемкий, и чтобы извлечь максимум из своего софта, вам понадобится мощная машина и терпение святого. И все же работа с *Valgrind* неизмеримо проще, чем многонедельный поиск причин периодического краха вашей программы, ведь *Valgrind* находит такие ошибки, которые больше никто не может отыскать. Для изучения инструментов вам понадобится только время: умения нужно не больше, чем для написания собственного кода. Несмотря на сложность вывода и многочисленность программ, включенных в комплект, программист средних способностей может многому научиться, пропуская свои работы через *Valgrind*, а для продвинутого программиста это «брат по разуму». LXF

Свойства навскидку



KCachegrind
Чем больше прямоугольник функции, тем больше процессорного времени она съедает

Библиотечные профили
Можно отфильтровывать лишние ошибки с помощью загружаемых профилей.



Вердикт

Valgrind 3.2.1

Разработчик: Julian Seward и Co
Сайт: http://valgrind.org
Цена: Бесплатно под GPL

Функциональность	9/10
Производительность	7/10
Простота использования	6/10
Документация	8/10

» *Не бойтесь его сложности. Valgrind – фундаментальный комплект отладочных инструментов.*

Рейтинг 8/10

Сравнение



Каждый месяц мы анализируем для вас тысячи программ – а вы можете отдохнуть!

Web-браузеры



В этом месяце **Ричард Драммонд** облачается в гидрокостюм и прихватывает несколько web-браузеров Linux для серфинга.



Paul Blachford

О тесте...

Каждый браузер мы прогоняли в присущем ему темпе по ряду сайтов, интересных для средних пользователей. Мы также тестировали дополнительные функции, например, чтение новостных лент, средства блокирования JavaScript или всплывающих окон и расширяемость. При выставлении рейтинга не принимались во внимание «непрофильные» функции, вроде встроенного почтового клиента или web-редактора.

Используемое оборудование: машина с процессором AMD 1350 МГц и 1 ГБ ОЗУ под управлением Debian Unstable. Мы использовали официальные сборки тестируемых браузеров (в случае *Firefox*, *SeaMonkey*, *Opera* и *Amaya*); при их отсутствии, если были доступны Debian-сборки текущих версий (*Konqueror*, *Galeon* и *Links2*), использовались они; если не было ни того, ни другого, мы сами собирали программу из исходных текстов (*Epiphany*, *Dillo* и *Lynx*).

Наш выбор

Web-браузер, вероятно, важнейший инструмент современного рабочего стола. Он больше не используется лишь для получения и просмотра документов с некоторого удаленного компьютера – это средство для предоставления пользовательских интерфейсов множества распределенных приложений, позволяющих нам работать с магазинами и банками, общаться, играть и учиться. Выбор браузера способен непосредственно повлиять на ваше восприятие этого богатого, нового сетевого мира.

Linux изобилует множеством браузеров на ваш выбор; по функциональности и возможностям они простираются от быстрых и экономных, чисто текстовых вещиц до высокотехнологичных произведений компьютерного искусства. Даже сегодня люди со скромными потребностями могут видеть смысл в использовании простого текстового браузера – в зависимости от мощности используемого оборудования.

Стандарты и безопасность

Соответствие браузеров стандартам – наиболее обсуждаемая тема сегодня, и это справедливо. Именно стандарт позволяет нам забыть кошмарные дни, когда разработчики делали сайты для *Netscape* или *Internet Explorer*... но не для обоих сразу. Однако web-разработчики придерживаются того, как браузеры интерпретируют стандарты, а не самих стандартов. И все еще можно найти сайты – в основном из разряда электронной коммерции, упорно отказывающие в соединении неизвестным браузерам. В зависимости от ваших потребностей, использование популярного браузера может оказаться практичнее, чем использование правильного браузера.

Помимо способности видеть сайт таким, как хотел разработчик, важно знать, что просматриваемый сайт действительно тот, о котором вы думаете, а не похожая на него подделка, созданная, чтобы выудить у вас конфиденциальную информацию. Безопасность – главная забота, когда осмеливаешься окунуться в хаос, которым является Интернет, и правильный выбор браузера может укрепить вашу защищенность. Многие браузеры предлагают инструменты для блокирования злонамеренных скриптов или незваных всплывающих окон, а некоторые даже предлагают средства защиты от фишинга и пытаются предупредить вас, если вы угодите на заведомо поддельный сайт.

Konqueror

Невоспетый герой проекта KDE.

Это куда больше, чем просто браузер и файловый менеджер. Благодаря мощной технологии компонентов KDE (*KParts*), *Konqueror* может встраивать интерфейсы самых различных приложений в одно из своих собственных окон. Это также средство просмотра изображений и PDF, оболочка к *CVS*, да все, что угодно (заметьте, что, поскольку он является частью KDE, это единственный браузер из Сравнения, отсутствующий на нашем диске).

Здесь же нас интересуют его возможности как браузера. Первые впечатления? HTML-движок от KDE – *KHTML* – радикально улучшился за последние несколько лет. Фактически, он перегнал *Mozilla Gecko* по скорости, потреблению памяти и соответствию стандартам. До недавнего времени он не получал достойного признания. Все изменилось, когда Apple портировала *KHTML* на Mac OS X и использовала его как основу для своего комплекса *WebCore*. Затем Nokia портировала его в *GTK* как *GtkCore* для использования в своих интернет-планшетах.

Konqueror работает с подавляющим большинством сайтов, а некоторые проблематичные сайты можно заставить сотрудничать, если «попросить» *Konqueror* притвориться *Mozilla*. Это легко делается в менеджере настроек, предоставляющем огромный выбор опций для настройки поведения *Konqueror*. Приложения KDE порой имеют репутацию чересчур настраиваемых, но мы нашли настройки *Konqueror* довольно простыми в использовании, и они управляют его огромной мощью.

Единственный недостаток *Konqueror* – он не доступен как самостоятельный браузер; но это расплата за тесную интеграцию с рабочим столом KDE. Программа запускается довольно хорошо и на Gnome, но без предварительно запущенного KDE старт будет слишком медленным. Обновление также намного сложнее, чем для любого другого браузера. Это просто позор, что никто серьезно не работает над браузером, основанным на *GTK*Core, чтобы мощь движка *KHTML* можно было испытать без потребности в KDE.

> «Неплохо запускается даже на рабочем столе Gnome, но тормозит там при старте.»



› В *Konqueror* не только превосходный HTML-движок, но и масса уникальных возможностей, типа просмотра в разделенном окне.



Konqueror

Версия: 3.5.5
Сайт: www.konqueror.org
Цена: бесплатно под GPL

» Просто лучший браузер для пользователей, работающих в KDE, и серьезный соперник на других рабочих столах.

Рейтинг 9/10

Opera

Альтернатива, «бесплатная как пиво».

Вслед за *Lynx*, *Opera* может претендовать на звание долгожителя среди тестируемых браузеров, уходя корнями в далекий 1994 г. Но *Opera* всегда была коммерческим продуктом. *Opera* Software пыталась получать прибыль от настольной версии своего браузера разными способами. Сперва он был условно-бесплатным, затем была выпущена бесплатная adware-версия. В наши дни *Opera* доступна для настольных платформ бесплатно без каких-либо ограничений.

Текущая версия *Opera* – интернет-пакет, включающий браузер, почтовый и IRC-клиенты и доступный для многих настольных систем. Компонент браузера также существует для популярных встраиваемых платформ, типа телефонов и приставок. Linux-версия собрана с *Qt*, и *Opera* поставляет двоичные файлы для впечатляющего множества дистрибутивов.

Opera всегда позиционировалась как быстрый браузер, и он быстр, хотя тестирование показало, что *Konqueror* может иметь некоторое преимущество в скорости отображения страниц. Однако хорошо реализованный интерфейс столь же важен для скорости, а *Opera* обладает множеством уникальных функций, делающих ее быстрой и удобной в использовании. Например, *Opera* первой стала поддерживать «жесты» мышью как дополнительное и очень гибкое средство управления. Она также располагает прекрасной функцией масштабирования, способной полностью изменить масштаб страницы, рисунков и всего остального – а не только шрифта. Это прекрасно, например, для людей с ослабленным зрением, и эта функция эффективно используется также в режиме предварительного просмотра печати и режиме «малый экран» (последний показывает, как будет выглядеть страница на встраиваемом устройстве).

Opera соответствует стандартам – это один из немногих браузеров, прошедших тест Acid2 (подробности см. на www.webstandards.org/action.acid2) – и корректно отображает подавляющее большинство сайтов.

> «Уникальные функции делают Opera быстрым и удобным браузером.»



› В *Opera* нет расширений а-ля *Mozilla*, но поддерживаются «виджеты» для создания апплетов типа Dashboard.



Opera

Версия: 9.02
Сайт: www.opera.com
Цена: бесплатно под закрытой лицензией

» Быстрый, мощный и лощеный браузер. Возможно, вы предпочтете его открытым конкурентам, Firefox и Konqueror.

Рейтинг 8/10

Galeon

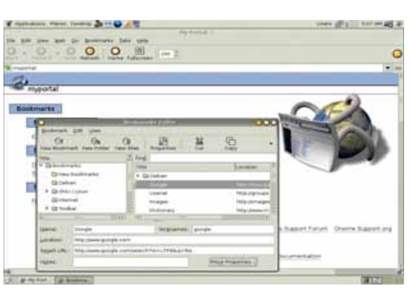
После ухода разработчиков в *Epiphany*, корабль-призрак?

F*irefox*, возможно, самый популярный «легковесный» браузер, основанный на *Gecko*, но он определенно не был первым. Эта слава принадлежит *Galeon*'у, вначале разработанному Марко Пезенти Гритти [Marco Pesenti Gritti] с целью: «сеть и только сеть». Вместо того чтобы взять на службу *XUL*, инструментарий пользовательского интерфейса проекта *Mozilla*, основанный на XML, *Galeon* использует более традиционный инструментарий *GTK*, стандартный во всех приложениях Gnome. Поэтому он ощущается более «родным» на рабочем столе Gnome и интегрируется лучше, чем «официальная» сборка *Firefox*.

Galeon переписан на *GTK 2.0*, но серия 2.0 пока не переняла всю ту функциональность, которая была в ранней серии 1.x. Инновационные «умные закладки» по-прежнему поддерживаются – здесь вы можете составлять собственные web-запросы, используя сайты типа Google.

Увы, развитию серьезно помешал уход разработчиков на ответвленный проект *Epiphany*, который занял место в качестве браузера по умолчанию в Gnome.

› Инновационные «умные закладки» *Galeon*-а.



Тем не менее, *Galeon* – довольно настраиваемый браузер, хотя и без поддержки расширений *Mozilla*, и пользователи могут предпочесть его более минималистичному *Epiphany*. К сожалению, он был заметно медленнее и потреблял больше памяти, чем любой другой браузер на базе *Gecko*, который мы тестировали.



Galeon

Версия: 2.0.2
Сайт: http://galeon.sourceforge.net
Цена: бесплатно под GPL

» Более настраиваемый, чем Epiphany, но вы, вероятно, выберете Firefox, если вам не нужна лучшая интеграция с Gnome.

Рейтинг 7/10

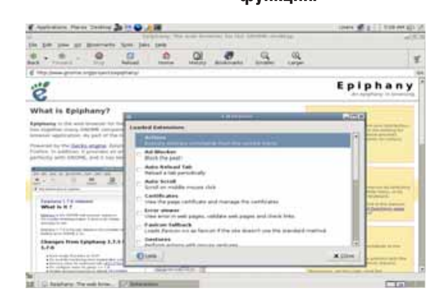
Epiphany

Новый браузер по умолчанию для Gnome.

Когда *Galeon* перерабатывался под *GTK 2.0*, разработчики разошлись во мнениях о направлении, в котором должен пойти проект. Основатель проекта Марко Пезенти Гритти хотел сохранить верность Gnome Human Interface Guidelines (HIG), но некоторые не желали упрощать *Galeon* до абсурда. Тогда Гритти создал новый проект, следующий букве HIG, и увел с собой значительную часть разработчиков *Galeon*'а.

Результат – *Epiphany* – вылитый *Galeon*. Это браузер на базе *Gecko*, с интерфейсом *GTK* вместо *XUL*. Возможно, основное отличие в том, что инструменты и опции настройки, которые не требуются среднему пользователю, не представлены в самом браузере. Согласны ли вы с этим аспектом философии дизайна Gnome – это отдельный разговор. К счастью, продвинутые функции доступны как дополнительные расширения. Они включают «умные закладки» *Galeon*'а и знакомую всем боковую панель. Обратите внимание, что расширения *Epiphany* отличаются от расширений *Mozilla*: *Epiphany* не способен использовать расширения *Firefox* непосредственно.

› Расширения восполняют большинство демонтированных функций.





Epiphany

Версия: 2.16.1
Сайт: www.gnome.org/projects/epiphany
Цена: бесплатно под GPL

» Прост в использовании и хорошо интегрируется в окружение Gnome, но, с другой стороны, Firefox, опять-таки, выглядит лучшим выбором.

Рейтинг 7/10

Dillo

Маленький размер, маленькое имя!

D*illo* – быстрый и легкий графический браузер, предлагающий полноправные, но лишь базовые web-функции. Он основан на версии 1.2 инструментария *GTK* и следствием этого является слабая интернационализация: например, отсутствует поддержка кодировки UTF-8. HTML отображается неплохо, но нет поддержки JavaScript или CSS, так что для сайтов посложнее *Dillo* не подойдет.

Впрочем, доступны заплатки сторонних разработчиков для добавления диалога настроек («официальный» *Dillo* нужно настраивать через текстовый файл), и для поддержки японской и русской локализаций.

Для тех, кто работает на скудном оборудовании, особенно на встраиваемых устройствах, *Dillo* по-прежнему прекрасный выбор. Используя уменьшенные панели инструментов, он предоставляет много пространства для просмотра, даже на устройствах с маленьким экраном.

Что ждет проект дальше? Разработчики вместо *GTK 2.0* выбрали для следующего поколения браузера не столь раздутый инструментарий *FLTK*. К сожалению, публичный релиз не выйдет, пока проект не найдет спонсоров. Похоже, что на рынке встраиваемых систем сейчас доминирует *Opera* – почти повсеместная на мобильных телефонах, а теперь даже на консолях Nintendo; и *Dillo* должен приложить много усилий, чтобы оправдать свое существование.

› Минимальные органы управления Dillo можно скрыть с помощью кнопки в нижнем правом углу.





Dillo

Версия: 0.8.6
Сайт: www.dillo.org
Цена: бесплатно под GPL

» Скудный, но только Lynx в этом обзоре использует меньше памяти, так что он может вам подойти, если вас раздражает Opera.

Рейтинг 6/10

Firefox

Жемчужина из короны Mozilla Foundation.

Мало кто из пользователей не слышал о браузере, известном как *Firefox* (у него было много имен за его короткую жизнь) – сейчас это любимец ИТ-изданий, а под Windows он привлек впечатляющее количество пользователей. Но так ли он хорош? Или мы просто сочувствуем проигрывающим, особенно тем, чья популярность растет, или после Internet Explorer 6 все кажется хорошим?

На самом деле, *Firefox* – полнофункциональный браузер. Он не самый впечатляющий в техническом отношении, не самый быстрый и не лучше всех поддерживает стандарты, но среди всех браузеров Linux он предлагает наибольшую полноту работы в сети. Действительно, нужно сильно постараться, чтобы найти сайты, не работающие с *Firefox*, конечно, исключая требующие проприетарных расширений, недоступных для Linux.

Вдобавок у *Firefox* легко расширяемый интерфейс пользователя со встроенной блокировкой всплывающих окон и средством чтения новостных лент. Активное сообщество постоянно разрабатывает дополнения, более чем восполняющие какую бы то ни было нехватку функциональности *Firefox*. На момент написания статьи приближался выход *Firefox* 2.0, и мы тестировали третий релиз-кандидат. Он основан на движке *Gecko* 1.8.1 (*Firefox* 1.5 использует версию 1.8.0), который не предполагает каких либо улучшений, кроме небольшого увеличения скорости и поддержки JavaScript 1.7. Основные изменения произошли в интерфейсе.

Firefox 2.0 включает множество новых функций, создающих комфорт при путешествиях по сети. Работать с вкладками стало удобнее, стало легче добавлять поисковые машины, поля для ввода текста теперь снабжены проверкой орфографии, и есть новый менеджер дополнений для управления темами и расширениями. Еще одна новинка *Firefox* 2.0 – защита от фишинга, работающая с базой выявленных фальшивых сайтов. Если вы зайдете на один из них, *Firefox* высветит заметное предупреждение, сообщающее вам, что этот сайт находится под подозрением.

> «Так ли он хорош... или просто после IE все выглядит хорошим?»



› **Новый менеджер дополнений упрощает установку и использование тем и расширений в *Firefox 2.0*.**



SeaMonkey

Mozilla Internet Suite восстал из мертвых.

После нескольких лет разработки интегрированного комплекса интернет-приложений, в 2003 г. в *Mozilla* Foundation, похоже, решили, что им больше нравится разрабатывать отдельные программы, так что *Mozilla 1.7* была последней в линейке. Да здравствуют *Firefox* и *Thunderbird*! Многие пользователи интегрированного комплекса – особенно корпоративные – стали горевать, что их бросили на произвол судьбы, и сообщество *Mozilla* замахнулось на создание нового проекта, продолжающего разработку *Mozilla*... но он больше не *Mozilla*.

«Что в имени тебе моем?» Будет ли браузер с новым именем столь же мил? Надеемся, что да, потому как Mozilla Foundation становится довольно педантичной по части своих торговых марок. Мало того, что она недавно повздорила с проектом Debian из-за распространения открытого *Firefox* с неодобренными изменениями (Debian отпочковал браузер под именем *IceWeasel*, чтобы обойти эти претензии), но даже этот проект – продолжение *Mozilla* Internet Suite – нельзя назвать *Mozilla*. Пришлось окрестить его *SeaMonkey*, старым кодовым именем комплекса.

Зачем выбирать комплекс, если есть более современный, более изящный и популярный *Firefox*? Хороший вопрос. С точки зрения функций, *SeaMonkey* полностью совместима с *Firefox* 1.5, да и построена на том же движке *Gecko* 1.8. Она отображает сайты точно так же, и предлагает те же самые «фишки», включая блокировку всплывающих окон и расширения.

Но вправду ли *Firefox* быстрее и эффективнее, чем полнофункциональный комплекс? На самом деле, нет. *Firefox* 2.0 чуть быстрее, но если говорить о потреблении памяти, вы не сможете просунуть между ними лезвие ножа. *SeaMonkey*, вероятно, даже менее громоздка, чем *Firefox* плюс отдельный почтовый клиент.

> «Сообщество Mozilla замахнулось на продолжение разработки.»



› **Внешне *SeaMonkey* неотличим от старого доброго *Mozilla Suite*.**



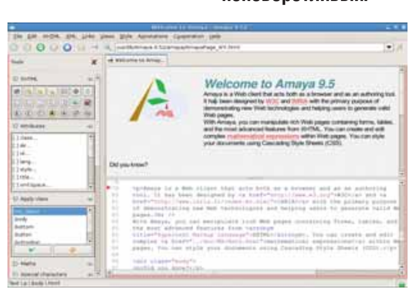
Amaya

Редактор, который, оказывается, еще и браузер.

Тим Бернерс-Ли [Tim Berners-Lee] считал, что сеть должна быть средой взаимодействия, и поэтому первый написанный им браузер – WorldWideWeb на базе Next – мог плавно выполнять редактирование, а также и просто отображать HTML-документы. Однако этот аспект сети никогда не отражался в основных браузерах и был забыт на долгие годы. Впрочем, один браузер следует заветам Бернерса-Ли: это *Amaya*.

Amaya разработана консорциумом W3C, поддерживающим web-стандарты, и используется скорее как тестовый полигон для новых web-технологий, а не как эталонная реализация браузера. Несомненно, проект сосредоточен больше на редактировании, чем на просмотре. *Amaya* поддерживает HTML, XHTML, MathML (для представления сложных формул в сети) и SVG, но CSS 2 – лишь частично. Более того, отсутствует поддержка JavaScript и даже закладок, а как браузер общего назначения *Amaya* слаба из-за необычного интерфейса. Поскольку *Amaya* еще и редактор, нечего удивляться, что она оказалась медленнее всех других наших браузеров, а уж как память-то ест...

› Можно редактировать страницы, но просмотр довольно неповоротливый.



Linux-версии можно собирать под *GTK* или *wxWidgets*. Интересно, что в последнем случае для отображения поддерживается OpenGL, но учитывая в целом низкую скорость *Amaya*, рекомендуем его отключить.



Amaya

Версия: 9.52
Сайт: www.w3.org/Amaya
Цена: бесплатно под лицензией W3C

» Удобный визуальный web-редактор с великолепной поддержкой MathML, но как браузер Amaya просто слаба.

Рейтинг 5/10

Lynx

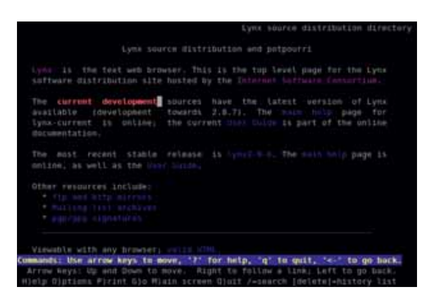
«Паутина» без картинок – на любом устройстве!

Гипертекст, конечно, возник раньше Всемирной паутины, и *Lynx* вступил в жизнь как клиент для проприетарного сервиса в Канзасском университете. Затем обрели популярность Gopher и WWW, и в 1993 г. появился *Lynx 2.0*, поддерживающий эти интернет-протоколы. С тех пор *Lynx* был портирован на десятки операционных систем, но придерживается своих корней, оставаясь чисто текстовым браузером. Он и сегодня находится в активной разработке.

К просмотру с помощью *Lynx* нужно попривыкнуть. Эта программа не пытается отобразить страницы способом графического браузера; вместо этого она оптимизирует использование пространства, доступного при ограниченной ширине устройства-терминала. *Lynx* не умеет отображать рисунки, но зато предоставляет выбор между показом имени рисунка, преобразованным в ссылку (и вы сможете просмотреть его с помощью внешней программы просмотра изображений), и полным его игнорированием. Поэтому сайты с табличным дизайном могут выглядеть довольно беспорядочно.

Если вы не падки до картинок и сможете поладить с интерфейсом, *Lynx* – эффективнейший способ просматривать подмножество ресурсов сети с любого устройства.

› Чисто текстовый просмотр гибок, но к нему нужно привыкнуть!





Lynx

Версия: 2.8.6
Сайт: http://lynx.isc.org
Цена: бесплатно под GPL

» Быстрый и эффективный браузер и лучший Gopher-клиент, но в наши дни полезен разве что если у вас нет графического терминала.

Рейтинг 6/10

Links2

Текст или графика? Выбирать вам...

Links начинал как чисто текстовый браузер, подобно созвучному с ним *Lynx*. Различные ответвления добавляли поддержку графики и экспериментальные функции, и в итоге объединились в проект *Links2*. *Links2* имеет текстовый и графический режимы (графический режим может работать в окне X, на framebuffer-устройстве в Linux или через SVGAlib), и частично поддерживает HTML 4.0 и JavaScript.

В отличие от *Lynx*, *Links2* отображает страницы – в частности, таблицы – так, как это делают «большие дяди», и его интерфейс поддерживает мышь, меню и диалоги. Оно и привычнее.

Разработчики *Links* упаковали впечатляющее множество функций в небольшой пакет, включив даже интерпретатор JavaScript. Но он отнюдь не так полезен, как кажется. Например, не поддерживается вызов скрипта как обработчика нажатия кнопки. Хотя *Links2* можно настроить через серию простых диалогов, при полном отсутствии документации не всегда понятно, что означают эти опции.

› *Links2* настраивается через не всегда интуитивные диалоги.



Нам всем случалось лихорадочно искать что-то в Google через текстовый браузер, и в этой ситуации нет ничего более подходящего, чем *Links*.



Links2

Версия: 2.1 pre23
Сайт: http://links.twibright.com
Цена: бесплатно под GPL

» Более удобный просмотр, чем текстовый режим Lynx, но графический режим действительно полезен лишь для тех, у кого нет X-дисплея.

Рейтинг 6/10

Web-браузеры

Вердикт

Firefox 9/10

Пусть *Firefox 2.0* не особенно быстр и эффективен в расходовании памяти, не справляется с тестом Acid2, но все же он – лучший универсальный браузер.

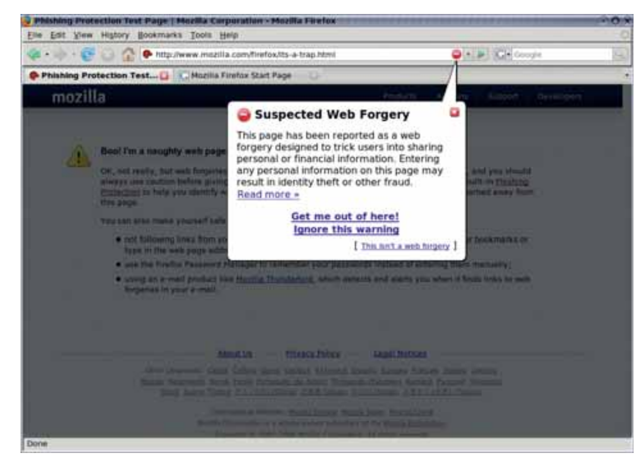
Помимо общей способности бродить по страницам, он обладает простым и понятным пользовательским интерфейсом и поддержкой массы сторонних расширений, добавляющих новые функции. Это также Linux-браузер, лучше всех поддерживаемый разработчиками, и его легко устанавливать и поддерживать в актуальном состоянии с обновлениями безопасности. Mozilla Foundation предлагает двоичные файлы для скачивания и серьезно относится к исправлению дефектов безопасности. *Firefox* снабжен встроенной функцией, позволяющей автоматически скачивать и устанавливать новые версии.

Тем не менее мы чуть было не вручили лавры победителя браузеру *Konqueror*. *Konqueror* заметно быстрее и легче, чем *Firefox*, и, вероятно, лучше следует стандартам. Однако сила *Konqueror* – его тесная интеграция с рабочим столом KDE – одновременно и его слабость. В наши дни KDE и Gnome взаимодействуют намного лучше, но установка всего KDE только ради браузера – сущий ад. *Konqueror* также медленно запускается, если KDE еще не загружен, и его сложнее поддерживать в актуальном состоянии по обновлениям безопасности, чем автономный браузер.

Под напором волн Web 2.0 становится все сложнее бороздить просторы сети с устаревшим браузером. Но если вы не любитель сайтов, требующих CSS или JavaScript, и ищете небольшой браузер, попробуйте *Dillo*. Он предлагает удобные способы работы в сети, быстр и хорошо работает на экранах с низким разрешением. Вообще без графики, *Links2* предпочтительнее, чем *Lynx*, поскольку отображает страницы более привычным способом. LXF

Ваше мнение

Является ли соответствие стандартам самой важной функцией браузера? Вам действительно нужны инструменты безопасности, типа защиты от фишинга? Текстовые браузеры без CSS и JavaScript все еще полезны сегодня? Напишите нам свое мнение на letters@linuxformat.ru.



› *Firefox 2.0* предупреждает: вы зашли на сайт, предположительно являющийся поддельным.

Таблица возможностей

	Amaya	Dillo	Epiphany	Firefox	Galeon	Konqueror	Links2	Lynx	Opera	SeaMonkey
Версия	9.52	0.8.6	2.16.1	2.0 rc3	2.0.2	3.5.5	2.1 pre23	2.8.6	9.02	1.0.5
Движок	Amaya	Dillo	Gecko 1.8.0	Gecko 1.8.1	Gecko 1.8.0	KHTML 3.5.5	Links2	Lynx	Presto 9.02	Gecko 1.8.0
CSS 2.1	Частично	☒	✓	✓	✓	✓	☒	☒	✓	✓
JavaScript	☒	☒	✓	✓	✓	✓	✓	✓	✓	✓
Прохождение Acid2	☒	☒	☒	☒	☒	✓	☒	☒	✓	☒
SVG	✓	☒	✓	✓	✓	✓	☒	☒	✓	✓
MathML	✓	☒	✓	✓	✓	✓	☒	☒	☒	✓
Доп.модули Netscape	☒	☒	✓	✓	✓	✓	☒	☒	✓	✓
Расширения	☒	☒	✓	✓	☒	☒ [1]	☒	☒	☒ [1]	✓
Вкладки	✓	☒	✓	✓	✓	✓	☒	☒	✓	✓
Блокировка изображений	☒	☒	☒	По URL	По URL	Глобально	По URL	N/A	Глобально [3]	По URL
Блок. всплывающих окон	N/A	N/A	Глобально	По сайтам	По сайтам	По сайтам	N/A	N/A	По сайтам	По сайтам
Блокировка JavaScript	N/A	N/A	Глобально	Глобально [4]	Глобально	По сайтам	Глобально	N/A	По сайтам	Глобально [4]
Чтение RSS	☒	☒	☒	✓[5][6]	☒	☒ [7]	☒	☒	✓	☒ [6]
Скорость CSS (мс) [8]	N/A	N/A	809	872	1,256	355	N/A	N/A	425	1,121
Скорость JavaScript (мс) [9]	N/A	N/A	2,529	2,642	2,532	1,483	N/A	N/A	667	3,820
Потребление памяти (резидентной) [10]	54 Мб	5,616 Кб	34 Мб	31 Мб	31 Мб	25 Мб	6,136 Кб [11]	2,524 Кб	29 Мб	31 Мб
Потребление памяти (разделяемой) [10]	16 Мб	3,344 Кб	21 Мб	17 Мб	18 Мб	18 Мб	2,556 Кб [11]	1,720 Кб	14 Мб	18 Мб

[1] Но приложения могут быть встроены в окно браузера через Kparts. [2] Но можно разрабатывать апплеты как виджеты Opera. [3] GIF- и Flash-анимацию можно блокировать в пределах URL. [4] В пределах сайта с помощью расширения NoScript. [5] Как «живые закладки». [6] Но средства просмотра новостных лент доступны как расширения. [7] Но можно использовать внешний Akregator. [8] Миллисекунды. Среднее значение по результатам десяти запусков CSS Test Марка Уилтона-Джонса [Mark Wilton-Jones], см. www.howtocreate.co.uk/csstest.html . [9] Среднее значение по результатам десяти запусков Javascript Speed Test Патрика Кейна [Patrick Kane], см. http://celtickane.com/projects/jsspeed.php. [10] В момент отображения сайта www.linuxformat.co.uk. [11] В графическом режиме в X-окне.

Distrowatch



Ежемесячная сводка новостей дистрибутивов Linux.



ЛАДИСЛАВ БОДНАР
основатель, редактор, начальник и сотрудник DistroWatch.com.

Немного о битах

Два года назад я заменил мою основную машину на новую высокомощную брэнд-систему на базе AMD64. После опробования нескольких дистрибутивов Linux я временно остановился на неофициальном AMD64-порте Debian Sarge, и вынужден был признать, что запуск 64-битной системы Linux на рабочей станции не оправдывает всех конфликтов из-за несовместимости ПО, отсутствия подключаемых модулей для браузеров и кое-как работающих драйверов.

Прошло всего два года, а изменения просто невероятны. Недавно я установил версию Mandriva Linux 2007 Powerpack для AMD64, которая, к моей радости, оказалась удивительно хороша – особенно порадовала отличная интеграция 32-битных библиотек и двоичных компонентов в ее 64-битную основу. Эту гибридную систему было столь приятно использовать, что не будь заметного ускорения, я бы забыл, какой у меня процессор.

Каждый пятый

Признаком удобства использования сегодняшних 64-битных Linux-систем является очевидный рост пользовательской базы. Для последнего выпуска Fedora Core 6, монитор трафика зарегистрировал, что где-то 20% всего канала занято пользователями, загружающими 64-битное издание. Другими словами, один из пяти пользователей Fedora в настоящее время работает в x86_64 ветке дистрибутива.

Я думаю, что 64-битный Linux значительно улучшился за последние два года. Если ранее это была идея для фанатов технологии, а не обычных пользователей, то теперь это доступная и впечатляющая архитектура. Если вы задумали купить новый компьютер, то присмотритесь к 64-битному. Вы не пожалеете.

ladislav.bodnar@futurenet.co.uk

Trash /bin

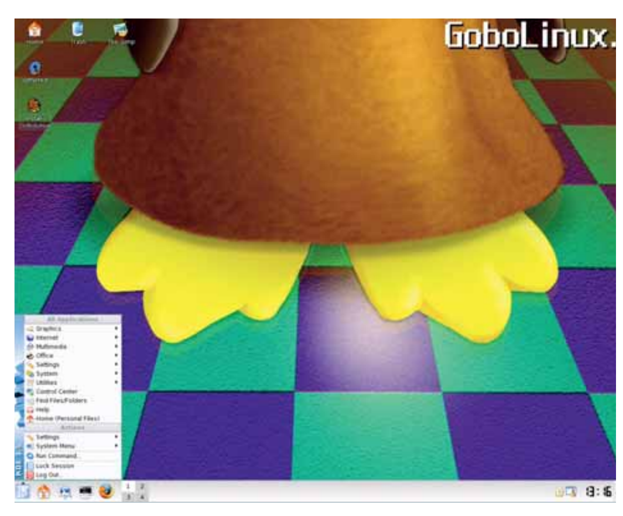
GoboLinux 013 Альтернативный дистрибутив с новой файловой системой...

Если загадочность (и замшелость) файловой системы Linux всегда приводят вас в замешательство, то пришло время пробовать GoboLinux. Этот проект-хобби, разработанный Хишамом Мухаммадом [Hisham Muhammad] и Андром Дечем [Andr Detsch], дерзнул покуситься на самую суть ОС Unix: ее 40-летнюю иерархию файловой системы. В GoboLinux вместо традиционных **/etc**, **/lib**, **/usr** и других корневых каталогов с загадочными именами, вы обнаружите более понятные **/System**, **/Files** и **/Users**. Разработчики GoboLinux создали действительно уникальный дистрибутив с более интуитивно понятной организацией файловой системы, чем в традиционных дистрибутивах Linux.

Проект зародился случайно. Пытаясь компилировать приложения без привилегий root, Мухаммад создал в своем домашнем каталоге папку **/Programs** и размещал в ней все вновь собранные программы, включая их файлы настроек в отдельных подкаталогах. Это оказалось весьма удобным решением, так что позднее, когда его жесткий диск сломался и он вынужден был переустанавливать ОС на новый, он решил собрать ее с нуля, основываясь на своем опыте размещения приложений в их собственных каталогах. Тогда-то, где-то в начале 2002, впервые и родился прототип GoboLinux.

С тех пор проект улучшил начальную структуру файловой системы настолько, что GoboLinux теперь пригодный к использованию дистрибутив Linux, а не испытательный стенд. При поверхностном взгляде он выглядит похожим на большинство других дистрибутивов: по умолчанию загружается в KDE и включает обычный набор свободных приложений. Но различия внутри, в том, что исполняемые и конфигурационные файлы расположены в совсем необычных местах. Например, для настройки менеджера загрузки Grub вам потребуется перейти в каталог **/System/Kernel/Boot/grub**, тогда как исполняемый файл Firefox и все его библиотеки можно обнаружить в **/Programs/Firefox/Current/**.

Истина в том, что оригинальная иерархия файловой системы в GoboLinux сохранена, но скрыта от глаз простых смертных при помощи заплатки ядра под названием GoboHide. Если вы загрузите другой дистрибутив с вашего жесткого диска и примонтируете раздел GoboLinux, то обнаружите традиционные для Unix каталоги **/etc**, **/usr** и другие. Это своего рода «страховочная сетка»: некоторые программы имеют пути, жестко прописанные в их коде, и иначе они не смогли бы нормально работать в «нестандартной» системе GoboLinux.



› **Рабочий стол GoboLinux выглядит как у всех...**

```
gobo@LiveCD /]ls -l
total 0
drwxr-xr-x   2 gobo gobo  40 2006-11-03 11:07 Depot
drwxr-xr-x   6 gobo gobo  40 2006-11-03 11:07 Files
drwxr-xr-x   7 gobo gobo 140 2006-10-29 11:32 Mount
drwxr-xr-x 200 gobo gobo  40 2006-11-03 11:07 Programs
drwxr-xr-x   6 gobo gobo  40 2006-11-03 11:07 System
drwxr-xr-x   3 gobo gobo  60 2006-11-03 11:08 Users
```

› **...зато в командной строке различия налицо.**

GoboLinux – отличный дистрибутив для опробования его в дождливый уикэнд. Он не для новичков в Linux, но включает несколько интуитивных графических инструментов, например, *GoboLinux Manager* для установки и обновления приложений. Он также предоставляет инструменты для компиляции и установки программ из исходных кодов одной командой и скрипт для создания собственных Live CD. Если вы где-то застряли, то разработчики будут рады помочь через список рассылки. И как знать, может быть, когда дождь перестанет, вы решите, что GoboLinux настолько занимательная ОС, что достойна занять постоянное место на вашем жестком диске.

www.gobolinux.org

Эпоха ренессанса

Mandriva Linux 2007 Омоложение проекта.

Mandriva Linux, некогда самый частый дистрибутив Linux на настольных компьютерах, пережил за последние годы драматический спад популярности: частично вследствие обычно запаздывающего свободно-загружаемого издания, но также вследствие появления более ориентированных на сообщество Ubuntu и OpenSUSE. Компания пересмотрела свой подход, и CD/DVD-образы последней версии Mandriva Linux 2007 выпущены одновременно с коммерческими изданиями дистрибутива.

Кроме того, 2007, кажется, добился хорошего качества. Объявлено, что релиз сфокусирован на недавно представленных 3D-эффектах рабочего стола, но у дистрибутива есть и другие тузы в рукаве. Одним из козырей является целостная интеграция 32-битных библиотек в его 64-битную основу. Это означает, что пакеты, скомпилированные не на 64-битной платформе, вроде чисто двоичных приложений *Opera*, *Skype* или *Adobe Reader*, легко будет установить в 64-битной системе Mandriva Linux 2007. Аналогично, 32-битные Java, Flash и медиа-расширения для браузера *Firefox* также доступны и установлены по умолчанию в издании Mandriva Powerpack.



› Mandriva One можно установить прямо с Live CD.

В свете недавнего соглашения Novell–Microsoft, приятно видеть, что возродившийся Mandriva теперь больше нацелен на сообщество. Если он и впредь будет идти таким путем, возврата прежней славы ему ждать недолго: Mandriva вновь завоюет сердца и умы многих пользователей настольного Linux.

www.mandriva.com

Очищение Ubuntu

gNewSense 1.0 ОС для пуристов.

Free Software Foundation (FSF – Фонд свободного ПО) жестко критикует дистрибутивы Linux, включающие (или упрощающие включение) проприетарных и несвободных компонентов. Делая это, доказывает Фонд, они снижают накал борьбы пользователей Linux за свое право на свободное ПО. Теряются также как минимум две из четырех важных свобод ПО, определенных FSF: свобода изменения исходного кода для приспособления к личным нуждам и свобода создавать улучшенные версии программы.

Чтобы угодить апологетам идеологии чистого ПО, фонд взялся поддерживать разработку полностью свободного дистрибутива под названием gNewSense. Это полностью свободное издание Ubuntu, в котором отсутствуют какие-либо проприетарные компоненты, зато добавлены некоторые GNU-приложения, например, текстовый редактор *Emacs* и важные библиотеки разработчиков. Заранее настроен доступ к обширному программному репозитарию Guniverse, и проект предоставляет обновления безопасности на время жизни продукта.



› Builder поможет изготовить производные от GNewSense.

gNewSense 1.0, основанный на Ubuntu 6.06, был выпущен в начале ноября 2006. Этот дистрибутив в настоящее время разрабатывается двумя ирландскими сторонниками свободного ПО: Брайном Бразилом [Brian Brazil] и Полом О'Молли [Paul O'Malley], но FSF набирает программистов-добровольцев, координаторов web-сайта, художников-графиков и разработчиков документации для помощи по части приложений и инфраструктуры проекта. Если вы верите в идеалы свободы ПО и следуете генеральной линии FSF, присоединяйтесь к gNewSense – прекрасный проект, хотя бы как база для нового дистрибутива! LXF

www.gnewsense.org

Плодовитые экспортеры дистрибутивов

Задумывались ли вы когда-нибудь, какая страна производит больше всего дистрибутивов? Избранные дистрибутивы, типа Debian или Gentoo, являются действительно международными проектами и имеют сотрудников во всех уголках мира; но многие другие создаются разработчиками-одиночками или малыми командами. А некоторые специально нацелены на конкретную географическую или языковую аудиторию, вроде бразильского Kurumin (*его талисман приведен ниже*). Таблица внизу содержит список стран, упорядоченный по количеству дистрибутивов, разрабатываемых на ее территории. Поищите сами на http://distrowatch.com/search.

Страна	Дистрибутивов	Примеры
США	79	Red Hat, Fedora, SUSE
Италия	20	Sabayon Linux, QiLinux
Германия	19	OpenSUSE, Knoppix
Бразилия	18	Kurumin, GoboLinux
Франция	18	Mandriva, Zenwalk
Испания	16	GnuLinEx, LinEspa
Канада	15	Xandros, VectorLinux
Великобритания	14	Blag, SmoothWall
Япония	12	Turbolinux, Vine
Нидерланды	9	Morphix, Nonux
Польша	9	Aurox, KateOS

Рисунок с обоев Лучиано Лоуренцо для Kurumin. Скриншот Карлоса Моримото.

Хит-парад дистрибутивов

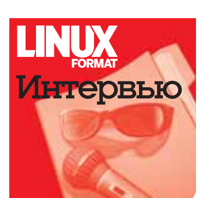
10 самых посещаемых страниц на DistroWatch.com с 11 октября по 10 ноября 2006 (среднее число визитов в день)

	Дистрибутив	Число визитов	
1	Ubuntu	2,920	‹›
2	SUSE	2,495	‹›
3	Fedora Core	2,239	‹›
4	Mandriva	1,086	‹›
5	SimplyMepis	953	↑
6	Debian GNU/Linux	857	↑
7	PCLinuxOS	825	‹›
8	Damn Small Linux	742	↑
9	Slackware	704	↓
10	Kubuntu	686	↑

» DistroWatch.com отслеживает популярность дистрибутивов, основываясь на количестве посещений сайтов, посвященных конкретным дистрибутивам. Хотя эти цифры и не отражают реальное количество инсталляций, они являются индикатором популярности дистрибутива на данный момент времени.

Время Gnome

Джефф Во отринул комфорт Canonical по зову рабочего стола GNU. И, как он сказал нам, занимается им ради своей мамы… и ради еще миллиарда пользователей, которым нужен FOSS.



Джефф Во [Jeff Waugh] славен своим вкладом в рабочий стол Gnome и своей ролью в стремительном успехе Canonical, где он работал над Ubuntu и развитием его сообщества вплоть до своего ухода в июле. Он – яркая личность, и мало найдется событий в мире свободного ПО и ПО с открытым кодом, которые обошлись бы без выступлений Джеффа. А еще его никогда не удавалось сфотографировать крупным планом. **Грэм Моррисон** и наш неустрашимый фотограф Джейсон не так давно пытались подловить его на OSCon. Но Джефф оказался слишком проворным. И все же команде LXF удалось заманить его в местный бар и побеседовать с ним о его пристрастии к разработке и развитию Gnome, о том, как ему живется после ухода из Canonical, и о том, как удалось Марку Шаттлворту [Mark Shuttleworth] привлечь к работе его команду преданных делу сотрудников.

LXF: В Gnome много отличных программ. Например, *Tomboy* и *F-Spot*…

ДжВ: Да, просто мы никогда не трезвонили об этом. Мы говорим об использовании Gnome на предприятиях, о том, как сделать его проще для пользователей, и о всяком другом, но в нем действительно много по-настоящему классных, радикальных функций и прочего, чем можно пользоваться.

LXF: А что вы считаете самой интересной частью Gnome?

ДжВ: Есть две-три вещи, по-моему, действительно интересные. Вы ведь знаете *Metacity*, оконный менеджер? В файле README *Metacity* Хэвок [Пеннингтон, лидер команды Desktop Team в Red Hat, – прим. ред.] описывает его, как Cheerios оконных менеджеров; для меня это сравнение культурологически невразумительное, но я так полагаю, что Cheerios – пресная каша! Говорится, что другие оконные менеджеры подобны Froot Loops и...

»

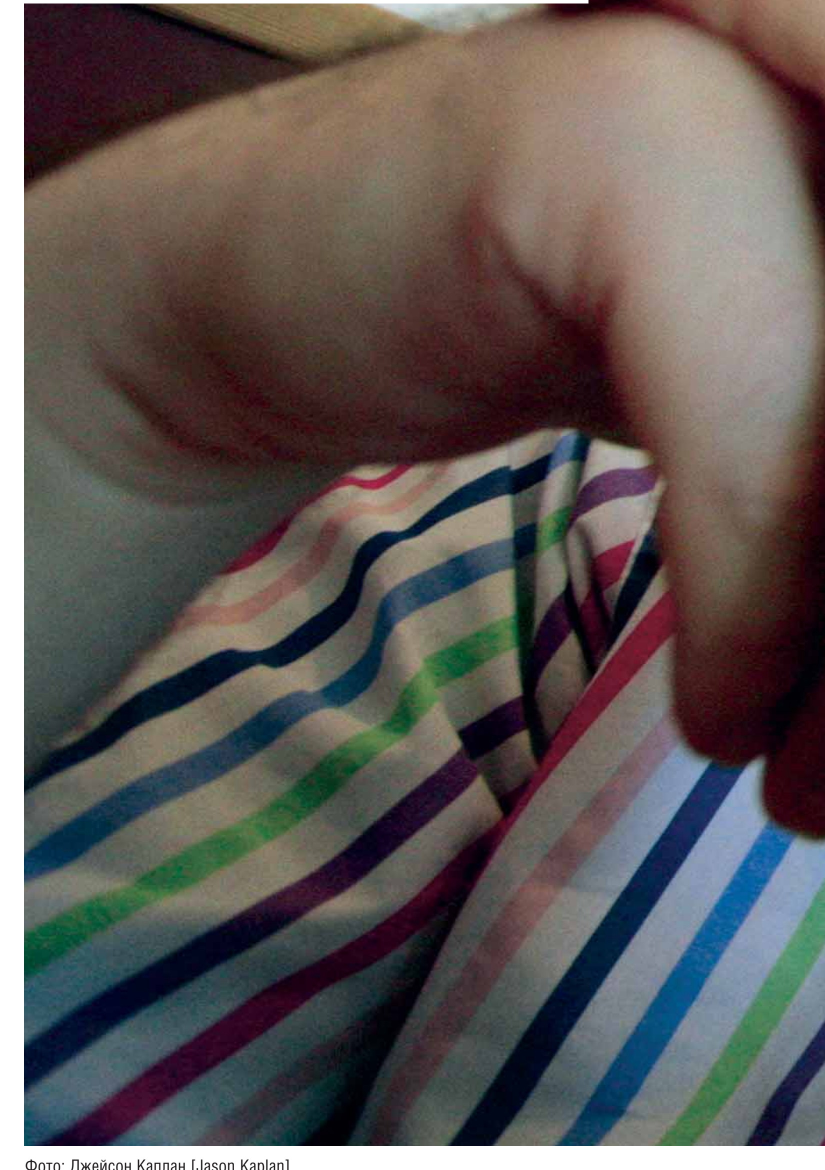


Фото: Джейсон Каплан [Jason Kaplan]



О РАННИХ ДНЯХ UBUNTU

«Марк как раз плыл в Антарктиду на ледоколе, и у него с собой были архивы рассылки Debian за полгода...»

LXF: Догадываюсь, что такое Froot Loops. [Американец-фотограф Linux Format Джейсон объясняет англичанину ГМ и австралийцу ДжВ, что Froot Loops – это хлопья с «искусственными красителями, сахаром и мерзким вкусом».]

ДжВ: Так вот девиз *Metacity* «Управляйте моими окнами и не путайтесь под ногами – знать не хочу про всякие там менеджеры окон; просто делайте это». Именно на это нацелен Gnome. Он просто работает, неважно как. Однако там есть крутая штучка под названием *Devil's Pie* [«Дьявольский пирог», – прим. ред.], этакий дополнительный пустячок. Подключите его – и сможете полностью автоматизировать работу *Metacity* сценариями на Lisp.

LXF: Звучит заманчиво.

ДжВ: Есть еще одна классная штука – *Brightside*, она заставляет стороны и углы вашего рабочего стола реагировать на разные действия. Например, если вы, перемещая окно, дотащили его до границы экрана, оно всплывет на соседнем [виртуальном] рабочем столе, вот такая навигация.

LXF: Как вы думаете, *Xgl* и *Compiz* будут встроены в оконный менеджер?

ДжВ: *Compiz* – интересная разработка. Наигравшись с композиционными менеджерами, все поняли, что вся проблема композиции настолько тесно связана с проблемами управления окнами, что ее нужно включить в оконный менеджер. Конечно, можно работать с ними и по отдельности, но тогда постоянно приходится осуществлять обращения то в одну сторону, то в другую.

LXF: Удивительно, что это работает.

ДжВ: Да. Но это действительно здорово. В *Compiz* есть одна штука... Итак, в *Metacity* тоже есть менеджер композиций, его встроили в существующий менеджер окон и добавили композиционные функции. А для *Compiz* создали прекрасный композиционный менеджер с возможностью расширений, и одно из расширений работает с окнами. В области управления окнами у *Metacity* успехи больше, но ведь за ним – годы работы над ошибками. Все эти сумасшедшие функции управления окнами добавлены в *Compiz* просто для выпендрежа ...

LXF: Вдруг все начали ими пользоваться.

ДжВ: Да, и никто не думал в стиле: «А давайте-ка сделаем работоспособный и правильный оконный менеджер», мысли были примерно такие: «Эй, вы, чокнутые из *Xgl*, мы тоже можем такое вытворять». Но *Compiz* внушает уважение, поскольку это своего рода технологический образец: его роль демонстрационная. Взять дрожащие окна: народу дурно делается, когда они начинают трястись, вот ведь как, но зато видны возможности технологии.

LXF: И далеко заводят возможности технологии *Compiz*?

ДжВ: Мирко Мюллер [Mirco Muller] поработал над созданием классной графики, используя Cairo, композиционные менеджеры и *Xgl*. Он приехал на Guadec [Европейскую Конференцию Пользователей и Разработчиков Gnome – Gnome User and Developer European Conference] в этом году в первый раз, и встретился там с Дэвидом Ривменом [David Reveman], Китом Паккардом [Keith Packard], Карлом Вортом [Carl Worth]... со всеми, кто создавал технологию, которую он использует, делая всякие красивые вещи. Ну, они начали разговаривать, уселись там, и поехало: «О, а еще надо бы сделать так! И вот этак!», разные крутые планы. Он, небось, пришел вечером домой и говорит [имитирует восхищенный голос]: «Ух, просто невероятно! Я толковал с Карлом и о том, и об этом, и обо всем…». А потом он принялся программировать, и создал такие замечательные вещи, как трехмерный вращающийся куб с анимациями Cairo на прозрачных гранях: теперь на вашем рабочем столе все эти Cairo-мультяшки крутятся на прозрачном кубе и видны все одновременно. Там есть GL-рендеринг, но все это еще и прозрачно, благодаря вашему композиционному менеджеру и применению Cairo для рендеринга. Все это полностью сглаженное, графика векторная и очень быстрая – действительно быстрая. Вот сведите таких людей вместе – и они понаделают чудес.

Разве не забавно: как-то вечером мы заговорили о его работе по клонированию дока Mac OS X. И Мирко сделал множественное отображение для значков, чтобы они выглядели одинаково четко при любом увеличении.

LXF: Значки используют SVG?

ДжВ: В том-то и суть. Сами по себе значки – SVG, но поскольку прямого пути между SVG и GL у нас нет, он конвертировал их в растровые изображения, и по мере увеличения их масштаба, он делал для них рендеринг в разных размерах.

LXF: Здесь-то и показали себя многоуровневые текстуры?

ДжВ: Да. Я говорю: «Значит, ты можешь отрисовывать SVG как GL?», потому что математически это почти одно и то же. И вот мы стали обсуждать, а что если взять SVG, превратить его в объект GL, и обработать его, и поиграть с результатом на экране – вместо мип-мэппинга текстур. Мирко похмыкал, поахал и призадумался, что бы можно создать таким образом. Потом мы поговорили обо всем этом с Китом и Карлом, и Кит объяснил, что лучше всего это делать с помощью *X*. Еще до *Xgl*, потому что *Xgl* – это *X*-сервер, являющийся клиентом GL и отображающий все как GL. У *X* была поддержка такой штуки под названием Glx… запутаться можно!

> **О философии Gnome**
> «Основополагающее значение имеет то, что мы изменяем восприятие ПО людьми.»

LXF: Однако вы привыкли.

ДжВ: Ну, да. [*Смеется*.] Glx – это расширение, позволяющее отправлять команды GL через X. Так что если у вас где-то там есть тонкий клиент и машина, вы можете здесь запустить программу, использующую GL, причем с аппаратным ускорением. Оказалось, что именно так и нужно создавать все эти прикольные GL-штуки. То есть вы трансформируете SVG в GL **здесь**. А Кит и говорит: можно сделать еще лучше – превратить это в макрос GL **там** и отправить объект и прочее как макрос через сервер – неважно, локальный или сетевой. И вы получите макрос GL, обрабатываемый аппаратно, его можно настроить. Фактически, мы усовершенствуем сетевой рендеринг, используя GL. Представьте себе аккуратненький, крошечный тонкий клиент с прекрасным графическим чипом Intel и всеми прочими наворотами, который через сеть выполняет работу, требующую аппаратного ускорения, причем экономит трафик.

LXF: Вижу, Gnome прошел долгий путь с релиза 1.0.

ДжВ: Да, 2.0 [вышедший в июне 2002] стал поворотным пунктом в нашей философии по части наших целей. На презентациях Gnome я говорил, что как группа мы выросли и повзрослели, и осознали действительно глубокий философский смысл того, что уже создано, и что свободное ПО предназначено не только для технарей.

Основополагающее значение имеет то, что ценности, обретенные благодаря свободному ПО, и тот факт, что наша «большая» свобода – свобода собраний, свобода слова – защищены нашим использованием свободного ПО в качестве инструмента. А моя мама этого не понимает. Она не видит угрозы, которую может представлять собой технология, и не видит, что мы отстаиваем наши свободы. Это очень важно, что мы можем изменить восприятие людьми программ и изменить их ожидания от программ. И в проекте Gnome это основной философский принцип.



Джефф – своего рода знаменитость в мире FOSS, и его постоянно приглашают на встречи и интервью.



LXF: Итак, почему же вы ушли из Canonical?

ДжВ: [*Смеется*] Ну, главным образом потому, что годика через три-четыре появятся дети (я только что отметил годовщину свадьбы). Я хочу рискнуть и заняться чем-то опасным – в какой-то степени – и если я не сделаю этого сейчас, то потом у меня уже не будет на это времени. У нас с женой был серьезный и важный разговор, и я сказал: «Вот что я думаю по поводу того, чем я хотел бы заняться, чтобы работать над Gnome несколько больше, чем раньше». Несколько лет назад у меня был период, когда я много работал как независимый консультант и получал приличные суммы, сделал сбережения, и потом месяцев девять изредка выполнял случайные работы, а почти полный рабочий день занимался Gnome.

LXF: Нельзя ли было при этом оставаться в Canonical?

ДжВ: Ну, это смешно. Многим разработчикам свободного ПО, работающим на фирмах, часто говорят: «Да тебе ж оплачивают полный день за занятие всякой ерундой!» Но на самом деле все не так. Я не думаю, что для Canonical будет разумно с финансовой точки зрения оплачивать мне полную занятость работой над Gnome. Нельзя забывать, что Canonical – небольшая начинающая фирма, они не делают деньги, во всяком случае, не сверхприбыли. Много средств они тратят на ShipIt [Canonical высылает диск любому, кто попросит об этом], да и на многое другое. Это маленькая и сплоченная команда, и у них нет таких уж супердоходов, чтобы с их стороны это было разумно [оплачивать мою работу над Gnome]. Я сам должен решить, как это делать!

LXF: А вас не поддержит Gnome Foundation?

ДжВ: Нет. На данный момент у Gnome Foundation все в полном порядке, и они зарабатывают кое-какие деньги; но на создание фондов и проработку финансовой модели, чтобы нанять сотрудников, нужно время. Так что я не особенно рассчитываю, что буду работать в Фонде. Может, когда-нибудь в будущем.

LXF: Как вы впервые вышли на Canonical?

ДжВ: Марк [Шаттлворт] позвонил моему другу, Роберту Коллинзу [Robert Collins], на LCA 2004 [linux.conf.au], проходившую в Аделаиде. Роб тогда работал над системой контроля версий *Arch* в TLA [исходный репозитарий *Arch*], а Марк знал, что он, в частности, собирался заниматься распределенным контролем ревизий, и это могло стать новой ступенью сотрудничества в мире Open Source. Марк поглядел на *Arch* и почуял, что сам *Arch* и идеи, которые стоят за ним, дают правильный взгляд на вещи. Поэтому он позвонил Робу Коллинзу на LCA, поболтал с ним и рассказал, чем хочет заняться.

В то время Роб был директором фирмы по консалтингу и разработке, и был вполне доволен своей работой, но Марк подкинул ему свою безумную идею, которая казалась неотразимой. Я как-то и забыл об этом, но сейчас вспоминаю, что Роб подошел ко мне на LCA и сказал: «Господи, только что у меня был поразительнейший телефонный разговор! Сейчас я не могу тебе рассказать, но когда-нибудь расскажу…»

LXF: Он был в космосе!

ДжВ: [Смеется.] Ну, в ИТ-индустрии такое постоянно встречается, так что я просто подумал: «Случилось что-то из ряда вон выходящее, но там посмотрим». На самом деле, Марк был на пути к Антарктиде…

LXF: Как вы…

ДжВ: [Смеется] – на ледоколе, и собой у него были архивы рассылок Debian за полгода.

LXF: Блестящая идея!

ДжВ: Да, потому что он знал, что в свободное время на этом ледоколе ему придется просто скучать, вот он и решил заняться чтением. А целью его было выяснить, чей вклад в Debian особенно велик, чтобы привлечь этих людей к себе. Нанимать сотрудников в мире Open Source замечательно просто – ведь вся работа на виду. На обратном пути он встретился с Робом в Сиднее, и Роб был очень не прочь [приступить к работе над *Arch* в Canonical на полный рабочий день]. Роб знал, что мои тогдашние задумки прекрасно вписываются в идею Марка. Я в то время работал на одного интернет-провайдера, где все было безнадежно, я уже созвал свою команду на обед и сообщил им, что у нас осталось 90 дней. Это было ужасно, просто в депрессию повергало. Ну, и Роб позвонил мне, и сказал: «Давай иди и встреться с этим парнем», а я говорю: «Да время не очень подходящее, лучше как-нибудь в другой раз». Но Роб настаивал, что это надо сделать сейчас, потому что этот парень пробудет здесь только один день; некоторое время мы с ним препирались, а потом он сказал: «Ну, хорошо. Ты когда-нибудь видел человека, побывавшего в космосе?» Я сказал: «Нет». «А хочешь увидеть?» «Да». «Тогда ступай и познакомься с ним, прямо сейчас. У него, похоже, есть для тебя работа». *Дзинь*! «Ну, раз так…»

> **О ПОТЕНЦИАЛЕ GL**
>
> **«Фактически, мы можем улучшить сетевой рендеринг, используя GL с тонким клиентом.»**

Ну, мы уселись, и Марк изложил свои взгляды. Первое, что он сказал – «Хочу создать дистрибутив Linux». Я чуть было не встал и ушел, потому что тогда казалось, что глупее этого ничего и быть не может. Но потом он начал объяснять, что за этим стоит, какова модель в его представлении, и какое это все имеет значение. Уже на тот момент у него было абсолютно ясное представление о таких вещах, как релизы, появляющиеся каждые полгода, рабочий стол Gnome, один CD, совершенно свободный для дальнейшего распространения, создание сетей поддержки, распределенный контроль за новыми версиями и создание условий для эффективного труда разработчиков дистрибутива, переход к модели, абсолютно не похожей на другие дистрибутивы Linux, и – это был один из основных пунктов – создание всего этого на базе Debian.

Вот он все это рассказал, а я и думаю: «Э, да ты действительно понимаешь, как эта модель работает и что творится в данной индустрии». И ведь через 90 дней я становился безработным! Роб-то был вполне доволен своей фирмой, и идея начинать с нуля казалась ему безумной. Это и вправду рискованно, и все, кто собирался работать в Canonical, навидались дот-комов [т.е. интернет-компаний, чьи сайты имеют расширение .com; после их бума последовали массовые банкротства, – прим. пер.]. Но Марк много чего наговорил, и убедил нас, что никаких дот-комовских глупостей не будет, а будет нечто долгоиграющее, сконцентрированное на основной идее и на коммерческой стороне; в общем, он не был просто эдаким папиком, он сам входил во все детали и занимался всем.

Он собрал сливки с Debian. Последние четыре релиза и тот рост Ubuntu, который мы наблюдаем за последние два года – результат работы 20 человек и их умения создать свое сообщество пользователей.

LXF: Я бы сказал, звучит впечатляюще.

ДжВ: А что касается коммерческого интереса, он тоже вырос… и наши бизнес-модели отсчитывали уже второй год, хотя сами мы еще были на первом. Многие смотрели на нашу модель, на что она похожа – и даже уже после первого предварительного релиза люди сказали: «Вот это нам и надо». LXF



Джефф защищает X, объясняет проблемы установки Dapper и болтает о своей теще в полной версии интервью на www.linuxformat.co.uk/waugh.html.



KDE 4

Вкус будущего!

Слухи о релизе v4 уже дразнят воображение фанатов KDE, но ждать его еще не один месяц. Ключевые разработчики KDE предоставили **Грэму Моррисону** предварительный обзор.

С момента выпуска KDE 3 в 2002 году, ландшафт Linux-технологий изменился до неузнаваемости. Теперь мы принимаем как данность поразительные 3D-возможности, интегрированный поиск и апплеты рабочего стола; и проектировщики KDE 2002 г. убеждаются, что их рабочий стол выдерживает перегрузки.

Но теперь не только Linux имеет вращающиеся кубы да виртуальные рабочие столы: новые версии Microsoft Windows и Apple OS X наращивают конкурентоспособность. Новая среда рабочего стола KDE должна будет сделать серьезную заявку на удобство Linux. Она должна стать доступной любому, простой в использовании и величавой на вид. Она должна переманить пользователей изуродованных «цифровыми правами» рабочих столов DRM в мир свободы, единения и сотрудничества.

До выхода KDE 4 все еще остаются месяцы, но дизайнерские планы нового рабочего стола утверждены, а библиотеки и спецификации уже доступны. *Linux Format* воспринял это как открытое приглашение сделать обзор. Мы оценили возможное влияние каждой технологии и поговорили с некоторыми из ведущих разработчиков, чтобы дать вам представление о вашем будущем рабочем столе KDE 4.

> «Рабочий стол KDE 4 должен быть доступен любому, прост в использовании и величав.»

Структура KDE 4

Oxygen
Plasma
Appeal
Phonon
Decibel
Solid
Пользователи KDE
Разработчики KDE

Структурная схема KDE 4 показывает, что пользователи получат хорошо спроектированный, ориентированный на задачи рабочий стол благодаря проекту Appeal, а появление Solid, Decibel и Phonon означает, что разработчикам больше не нужно будет изобретать велосипед при добавлении новейшего оборудования, мультимедиа и сетевых функций.

Манифест Appeal

Как мог бы сказать Стив Баллмер: конечные пользователи, конечные пользователи, конечные пользователи, конечные...

KDE порожден немецким компьютерщиком Маттиасом Эттрихом [Matthias Ettrich] в 1996 г. Он всегда строился на коммерческом продукте *Qt* от норвежской фирмы Trolltech. Но KDE, по большому счету, крупнейший проект, использующий *Qt*, и Маттиас Эттрих теперь директор Trolltech по разработке ПО, а значит, эти два проекта серьезно влияют друг на друга.

Qt – инструментарий программиста и костяк функциональности KDE, выполняющий многие трудоемкие задачи, например, манипуляцию изображениями и файлами, прорисовку элементов графического интерфейса и взаимодействие с базой данных. При выходе каждого основного релиза *Qt* KDE должен адаптироваться к изменениям и вбирать новые функции. Вот почему цикл выпуска KDE следует за *Qt*. *Qt 4*, вышедший в июне 2005 г., ввел много новых функций, которых не хватало KDE, в том числе, новый движок рендеринга Arthur и поддержку SVG.

Но команда KDE, обновляя версию, не упустила шанса сделать гораздо больше, чем просто пересмотр функций. Кроме замены устаревшего механизма взаимодействия процессов и принятия D-BUS, разработчики добиваются внешней привлекательности. При переводе внутреннего формата изображений на SVG и добавлении расширяемых виджетов, разработчики хотят, чтобы новые пользователи понимали среду рабочего стола интуитивно. Мэтт Роджерс [Matt Rogers], глава проекта входящей в состав KDE IDE *KDevelop*, говорит: «Я думаю, во главе угла для разработчиков KDE 4 сейчас находится интерфейс пользователя. Говорю это только потому, что слишком много было спекуляций, шумихи и прочего вокруг всяких интерфейсных технологий, типа Plasma и Oxygen». В KDE 4 пользовательский интерфейс становится «первым лицом».

Paul Blachford

Полигон идей

Экскурсию по KDE 4 удобно начать с Appeal. В отличие от технологий, которые мы рассмотрим дальше, Appeal [зов, – прим. пер.] – это не конкретный программный проект, а, скорее, исследовательский центр по «перетягиванию» KDE от разработчиков поближе к простым смертным. Не очень внятно? Дело в том, что Appeal – пока только манифест для задуманных в KDE 4 новых технологий, видимых пользователям: они должны сразить их в первую же наносекунду и заставить влюбиться в этот рабочий стол. Проект Appeal хочет, чтобы пользователи KDE 4 воспринимали функциональность так же, как Йода воспринимал Силу – просто как должное.

> «Пользователей нужно сразить в первую же наносекунду, чтобы они полюбили этот рабочий стол.»

Сайт http://appeal.kde.org был запущен в августе 2005 г., и его wiki переполнена идеями. Проект занимается «визуальными искусствами» KDE 4, включая Plasma (похожую на *SuperKaramba* замену виджетов рабочего стола, икон и панели) и Oxygen, гладкий, векторизованный набор иконок, определяющий внешний вид KDE. С некоторого момента Appeal займется и технологией поиска, которую будет использовать KDE 4. Аарон Сейго [Aaron Seigo], штатный разработчик Plasma, описывает Appeal как «социальный эксперимент по сознательному вовлечению в скоординированную, открытую и многоцелевую разработку». По сути, Appeal стал координационным центром для новой плеяды разрабатываемых приложений KDE 4.



› **Просмотрщик документов *Okular* – первое приложение KDE 4, написанное с нуля; оно следует правилам Appeal.**

Манифест Appeal содержит четыре принципа: «произведение искусства», «рай разработчиков», «сетевые вычисления» и «интегрированный рабочий стол». Каждый из них немного расплывчат, ибо Appeal не определяет требования: он только иллюстрирует идею. Поэтому «произведение искусства» стоит на первом месте. Подобно Марку Шаттлворту, разработчики KDE признают, что красота – это функция. KDE 4 в первую очередь нуждается в приятной внешности, вот почему дизайн иконок Oxygen непосредственно связан с Appeal. «Рай разработчиков» означает, что разработчикам должно хотеться работать с KDE API и при этом ощущать, что использование технологий KDE – лучший способ достичь их целей. Есть даже мысль создать в сети «университеты» KDE-разработки, где можно изучать KDE, чтобы максимально упростить новым разработчикам создание собственных проектов.

У «сетевых вычислений» более прикладная цель, движимая идеей единой регистрации для пользователя: и за получение почты, и за доступ к удаленному столу по протоколу NX, и за web-сервисы будет отвечать единственный механизм аутентификации. Это согласуется с инициативой KDE/Wikipedia, анонсированной в июне 2005 г.: разработкой интерфейсов KDE API для выполнения запросов к Wikipedia, которые в дальнейшем сможет использовать каждое приложение KDE.

Appeal помогает убедиться, что каждое приложение KDE 4 использует один и тот же подход к дизайну интерфейса. Удобство – ключевой момент, и это означает, что функциональность должна быть как можно более интуитивной: нет смысла дарить пользователю мощь KP*Arts* или KIO slaves, если он даже не подозревает о подарке. Вот как сильно разработчики KDE хотят, чтобы вы использовали их рабочий стол. »



Plasma и Solid

Не парьтесь с настройкой оборудования: лучше поиграйте с виджетами.



Plasma – «интегрированный рабочий стол» Appeal, и это первое, что бросится в глаза в KDE 4. Эффекты Plasma, в отличие от многих других новых технологий этой версии, вы сможете увидеть на рабочем столе.

Если вы не программист, то старый KDE 3 для вас нерасширяем: невозможно создавать собственные элементы интерфейса или добавлять функции к рабочему столу. Вы ограничены в использовании предоставляемых инструментов, и их недостаточно. Самое популярное приложение для расширения функциональности KDE 3, *SuperKaramba*, больше похоже на одномоментный «хак», чем на основу для расширений. Но это не вина автора прежней *Karamba*, Ганса Карлссона [Hans Karlsson], а следствие слабой поддержки расширений в KDE, отсюда и «хаки»: просто не было другого способа решить проблему. Plasma – средство KDE 4, призванное исправить этот недостаток. Это интерфейс к рабочему столу, способ добавлять расширения, не ломая машину.

Проблема Kicker

К сожалению, разработка Plasma зависит от некоторых других функции KDE 4, и потому она шла медленно. Лидер проекта, Аарон Сейго, до июля 2006 г. дожидался стабилизации новых API, достаточной, чтобы начать разработку, и многое изменилось в сентябре, после конференции Akademy в Дублине (ежегодная встреча KDE-сообщества).

Сейго имеет огромный опыт управления проектами типа Plasma, и к разработке Plasma его привела, прежде всего, его работа над KDE-панелью *Kicker*. *Kicker* часто воспринимается как некий «хак» (похоже, это общая тема приложений KDE). Он не предусматривал обработки прозрачности, кнопок или множественных апплетов, и в результате добавления этих функций исходный код стал неуправляемым и сложным в сопровождении. Были случаи, когда разработчики, хотевшие всего-навсего сменить цветовую схему, обнаруживали, что проще ответвить *Kicker* и решать свои задачи в этом коде, чем вносить свои изменения в исходный проект с тем же самым эффектом. Сайт http://kde-apps.org содержит некоторые ответвления *Kicker*, которые вносили в него только косметические изменения, не влияя на суть исходного проекта. *Kicker* даже научили автоматически перезагружаться, если он зафиксирует крах своего собственного процесса.

› **Временная шкала вверху экрана – это простой виджет Plasma, плод проекта Google Summer Of Code.**

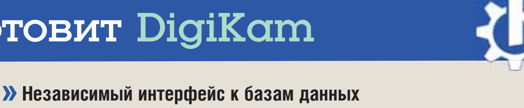
Частично эта проблема вызвана тем, что подавляющее большинство приложений KDE разработано на языке C++. Он очень мощный, но, по словам Сейго, «может и крышу снести». C++ слишком сложен для разработки простых расширений и расставляет слишком много ловушек для неосторожных программистов. *SuperKaramba* обходит эту проблему, используя язык Python, довольно простой по сравнению с C++, и подобное сделано и в Plasma. Plasma по-прежнему будет разрабатываться на C++, но расширения предполагается писать на JavaScript, в основном потому, объясняет Сейго, что JavaScript гибок и прост. Также планируется поддержка других популярных языков, включая Python и Ruby.

Создайте свой рабочий стол

Лично на вас окажет влияние тот способ, которым Plasma использует пространство рабочего стола. Plasma заменит метафору иконок тем, что лучше соответствует современным технологиям. Апплеты, подобные предлагаемым *SuperKaramba*, станут разменной монетой для рабочего стола, а компоненты типа Kicker или менеджера задач будут представлены отдельными апплетами Plasma. Дойдет даже до перетаскивания в апплеты компонентов из некоторых основных приложений. Например, перетаскивание папки входящих сообщений из *KMail* на ваш рабочий стол могло бы динамически создать, скажем, апплет оповещений о поступлении новой почты.

Вот тут-то и пригодилась идея KDE об абстракции данных, потому что если у *SuperKaramba* и есть проблема, так это недостаток согласованности – существует десяток апплетов для вывода информации о погоде или мониторинга системы, и каждый собирает данные по-своему. Отделение данных от интерфейса апплета будет означать, что как только будет создан внутренний механизм сбора информации о погоде или о системе, каждый, кто обладает минимальными художественными способностями, сможет легко добавить пользовательский интерфейс. Сложность здесь в разработке источников данных. Они будут многомерными (как база данных), и разрабатываются на C++. При этом каждый источник сможет иметь более одного атрибута – например, число писем в вашем почтовом ящике наряду с количеством писем от конкретного человека.

KDE 4 собирается свернуть менее успешные части KDE 3, а оставшиеся упростить. Иногда достаточно изменить соответствующий API, иногда приходится создавать новый, как в случае с Plasma.



Что нам готовит DigiKam

- **Независимый интерфейс к базам данных**
- **Интеграция с рабочим столом**
- **Поиск по метаданным**

Жиль Колье [Gilles Caulier], «хранитель» *DigiKam*: «После выхода версии 0.9.0 было начато портирование *DigiKam* на *Qt 4*/KDE 4. Первая задача новой библиотеки – предоставить новый интерфейс к базам данных, независимый от выбранного сервера (как в Amarok). С *Qt 4*, в отличие от *Qt 3*, это будет сделать легко. *Qt 4*/KDE 4 также будет управлять всеми поточными фильтрами изображений, доступными в ядре *DigiKam*. На самом деле, стиль кодирования потоков в *Qt 3* не особо удобный, особенно когда пытаешься получить обратную связь с графическим интерфейсом пользователя во время расчета или загрузки файла.

За последний год мы много работали над *DigiKam* и *DigiKamImagePlugins*, чтобы добавить поддержку 16-битного цвета, управление цветом, полную поддержку метаданных, поддержку файлового формата RAW и так далее – между 0.8.x и 0.9.0 разница очень большая. Исходный код использует только *Qt 3*/KDE 3, и сейчас он на стадии завершения. Выпуск 0.9.0 планируется на конец 2006 г. Фактически, у нас нет времени на изучение нового API, предоставляемого KDE 4. Этим мы займемся после выхода 0.9.0.»

Но есть несколько областей, нуждающихся в полном переосмыслении и разработке с нуля, например, медиа-проигрыватель. Solid и Phonon (*см. стр. 32*) – два новых API, спроектированных, чтобы сделать KDE максимально портируемым – и это особенно важно сейчас, когда свободная версия *Qt* доступна для операционных систем Mac OS X и Microsoft Windows. Как и в случае с Plasma, этот слой абстракции требует разделения на интерфейсную (front-end) и фоновую (back-end) части. Solid и Phonon отделяют данные от пользовательского интерфейса настолько, чтобы фоновый поставщик данных можно было менять, а интерфейсная часть продолжала бы работать. Все это – благодаря мощному механизму «сигналов» и «слотов» в *Qt*, дающих программистам свободу создавать интерфейсы в ПО.

Solid'ная конфигурация

Solid – это слой аппаратной абстракции, который KDE будет использовать в будущем. Он поможет забыть о настройке оборудования, и, будем надеяться, сотрет грань между приложениями для конкретных устройств. Лучший пример – беспроводные сети. При использовании Solid, информация о сетевом соединении и его состояние станет доступна любому приложению KDE, использующему Solid, причем ему не надо будет вникать в особенности оборудования. Это означает, что апплет Plasma, менеджер сети или даже ваш FTP-клиент будут способны перемещаться между беспроводными сетями или отслеживать статус соединения, не надоедая вам лишними запросами пароля.

В KDE 3 обнаружением и настройкой оборудования вынуждено заниматься каждое приложение. Например, для настройки K3b вам нужно сообщить, какие CD/DVD-приводы вы используете. Solid, в фоновом режиме, делает эту информацию доступной для всех остальных средств записи CD/DVD, и вам не нужно вводить одну и ту же информацию дважды. Это особенно важно для областей, где приложения для работы можно выбирать. Так, в KDE 3 есть два инструмента для работы с беспроводными сетями (*KNetworkManager* и *KWiFiManager*), а также различные приложения для определения оборудования, и их следует переработать в KDE 4, чтобы использовать преимущества аппаратной абстракции Solid. В этих случаях пользовательский интерфейс Plasma будет расположен поверх слоя абстракции Solid, и вам не придется каждый раз «изобретать велосипед».

Solid работает в так называемых доменах. Они сообщают максимум информации о затрагиваемом оборудовании (хотя непосредственно с ним не взаимодействуют). Есть домены для распознавания оборудования, управления питанием и управления сетью. Поверх доменов располагаются агенты политик (policy agent), занимающиеся характеристиками оборудования. В данный момент их всего два: один для управления сетевыми соединениями и второй – для обработки мультимедиа.

Может показаться, что технологии, подобные Plasma и Solid, сами по себе не новы, но огромным отличием KDE 4 является то, что о них думали до начала разработки какого-либо приложения, так что Plasma и Solid становятся доступны любому приложению KDE 4. Для разработчиков это упростит добавление функций во время портирования приложения на KDE 4 – только представьте, какой потенциал это дает для *KOffice*, *Kopete*, *Kate*, *Konqueror*. Изменится самый способ использования KDE.

> «В KDE 4 о новой технологии думают до того, как приступать к программированию.»



Что нам готовит K3b





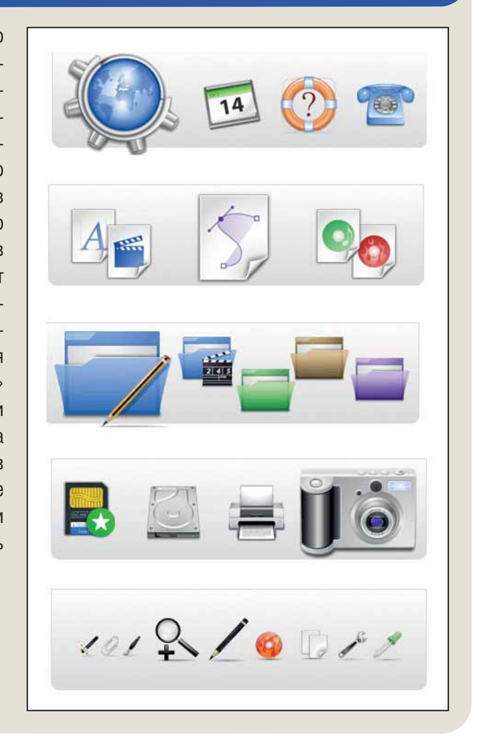
» **Однократная настройка устройств для всех приложений сразу**
» **Простая установка**
» **Интегрированный в систему поиск**

Себастьян Трюг [Sebastian Trueg], глава K3b: «Я уже говорил с разработчиком Solid об аппаратной абстракции, и мы обсудили возможность включить функции *K3b* непосредственно в Solid или даже в HAL. Сейчас я занят релизом 1.0 (который пришлось опять отложить). После этого начнется портирование на KDE 4.

Что касается поиска, я думаю, что интеграция в *K3b* будет такой же, как во всех приложениях KDE. По крайней мере, это моя цель: получить глубокую интеграцию расширенных семантических функций рабочего стола KDE. Я пытаюсь следить за всем, что творится вокруг, но это все-таки неподъемно. Так что я сосредоточился на проектах, над которыми работаю (*K3b* и Nepomuk-KDE) и пытаюсь следить за развитием родственных проектов.»

Оживляя иконки: Oxygen

Маркетинг KDE 4 весьма напорист, и это распространяется даже на иконки: разработчики продвигают Oxygen как «не просто очередную тему иконок. Oxygen формирует новые принципы работы пользователей». Но Oxygen – и вправду просто тема иконок; единственное ее отличие в том, что разработчики Oxygen не только перенесли привычный набор значков в масштабируемый векторный формат SVG, но и добавили иконкам выразительности, использовав стандартные цветовые палитры и фон для обозначения различных функций. Иконки «действий» типа Редактировать, Увеличить или Настроить выделяются на общем фоне, а значки выбора файлов расцвечены в соответствии с типом. Что-то похожее было в некоторых приложениях KDE и раньше, но никогда не распространялось на весь рабочий стол.



› **Масштабируемая графика будет играть важную роль во внешности и работе ПК с KDE 4 – даже если нет графического интерфейса.**

Phonon, KHTML и Decibel

Классные инструменты рабочего стола, понятные даже бабулькам.



Phonon перенимает то, что делает Solid для настройки сетей и управления питанием, и применяет то же самое разделение на фоновую и интерфейсную части для воспроизведения и записи мультимедиа.

Дело важное, если учесть, насколько отвратителен старый аудиокаркас из KDE 3. *Arts*, аудио-демон KDE, создает больше проблем, чем решает. Разрабатывался он с целью освободить программиста от мороки со входами и выходами кодеков и аудио-оборудованием, но в итоге получилось, что *Arts* редко ведет себя прилично с другими драйверами и может отказаться воспроизвести простой звук. Мультимедиа в Linux развиваются, появился, например, *GStreamer*, создающий стандартную платформу с низкой латентностью для воспроизведения звука. Phonon не привязывает воспроизведение к API, он просто предлагает программисту приемы работы. Воспроизведение должно работать прозрачно, независимо от того, запускает ли пользователь KDE-приложение на Linux, OS X или Windows.

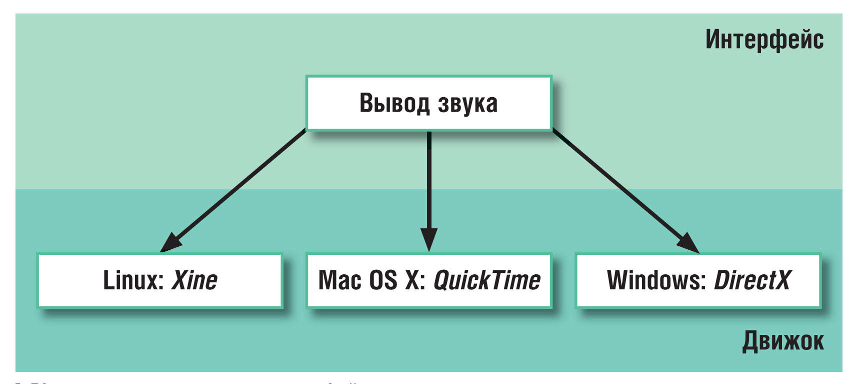
Отсоединение проигрывателя от кода, выполняющего воспроизведение, означает, что пользователю больше не нужно беспокоиться

> «Болтовня с друзьями станет так же привычна за рабочим столом, как управление файлами.»

о том, какой фоновый процесс настроен и как он будет взаимодействовать с другими приложениями. Разработчики Phonon надеются расширить этот принцип на самонастраивающиеся (plug-and-play) устройства, типа USB-гарнитур, где воспроизведение будет просто работать без всякой дополнительной настройки, и пользователь даже и вникать не станет, какой фоновый процесс этим занимается. Как и в Solid, фоновый процесс динамически загружается и легко заменяется, а Phonon продолжает работать.

Без GStreamer

У Phonon есть большая нерешенная проблема: не налажена взаимосвязь с самым популярным механизмом воспроизведения в Linux – *GStreamer*. Сейчас доступны только реализации для *Xine*, *AvKode* (интерфейс Phonon к библиотеке FFMPEG) и NMM (для сетевых аудиопотоков). Это действительно проблема, и без *GStreamer* Phonon, вероятно, ждет та же участь, что и его предшественника *Arts*. Даже самый сложный движок, *Xine*, поддерживает воспроизведение, но не запись; будем надеяться, что ко времени выхода KDE 4 эта проблема будет решена. Другие функции, которые, вероятно, будут включены в релиз – поддержка KIO slaves, VoIP на уровне API (облегчающее разработчикам приложений добавление интернет-телефонии), встроенное воспроизведение DVD и совместимость с DVB.



› **Phonon отделяет движок от интерфейса для воспроизведения мультимедиа.**

Что нам готовит Kate





» **Двунаправленный текст (поддержка языков типа арабского)**
» **Произвольная подсветка**
» **Автодополнение текста**

Кристоф Куллманн [Christophe Cullmann], глава Kate: «Интеграция с Tenor – это, наверно, круто, но для нее нужен код Tenor, а его пока нет. Мы будем поддерживать скрипты KPart, что позволит писать, например, средства расстановки отступов в коде как сценарии JavaScript. Графический интерфейс останется примерно тем же; кое-что переделано, например, диалоги поиска/замены перейдут в поисковую панель, вроде используемой в *Mozilla Firefox*. Самая большая проблема на данный момент – надежность; требуется масштабное тестирование и отлавливание ошибок в кодовой базе KDE 4.»

Прорыв KHTML в Mac

KHTML – оплот стандартов для проекта KDE. Это основной компонент рабочего стола, используемый для отображения web-страниц в самых различных приложениях. Он также принят как движок отображения HTML во многих популярных браузерах, включая *Safari* от Apple, открытый браузер от Nokia и, с недавнего времени, платформу Adobe *Apollo*.

Благодаря тесной интеграции с остальной частью KDE, *KHTML* зачастую отображает страницы быстрее, чем независимые браузеры вроде *Firefox*. *KHTML* всегда разрабатывался с прицелом на легковесность, и еще одним его преимуществом является возможность его повторного использования: это один из первых проектов, начавших использовать каркас KP*Arts* в KDE 2, и благодаря ему вы можете видеть web-страницы в приложениях, не имеющих непосредственной связи с HTML: Например, *Amarok* отображает страницы *Wikipedia* непосредственно в окне приложения. *KHTML* также является важной частью и *KMail*, и *Akregator*, наряду со многими другими приложениями. Именно портируемость обусловила успех *KHTML*.

Интегрированный интернет – не единственное сходство пользователей KDE и их «кузенов» с Mac: значительная доля *KHTML* используется для разработки платформы рендеринга *Safari WebCore*. Благодаря лицензии LGPL разработчики Apple возвращают все выполненные изменения в *KHTML* – в результате *Konqueror* оказался одним из двух браузеров, прошедших тест Acid2 в ноябре 2005 г. Но многие изменения, сделанные командой Apple, слишком далеки от текущего исходно-

го кода *KHTML*, чтобы объединение функций стало осуществимым.

Получается, что *Safari* хорошо работает со сложными сайтами, вроде календаря и почты Google, а *Konqueror* плетется позади. Выпуск KDE 4 – отличный повод переписать *KHTML*, чтобы он стал способен принимать код Apple. Эта инициатива получила наименование Unity, с целью «примазаться» к хорошим отзывам в прессе, которые получили другие браузеры на базе *KHTML*. Она также дает гарантию, что в будущем разработка *KHTML* будет согласована с основной ветвью разработки *Safari*. Это не только поможет с совместимостью *KHTML*, это также будет означать соответствие с точностью до ошибок коду Apple, дающее опору на многочисленных пользователей и улучшающее базу стандартов для принятия в браузере. Обязательство серьезное, потому что команда *KHTML* Unity собирается использовать код Apple как новую базу для выстраивания совместимости *KHTML*, а не реформировать текущую ветвь разработки *KHTML* – получается что-то вроде эстафеты, где эстафетная палочка возвращается первому бегуну. Но есть несколько недостатков. Может пострадать производительность браузера, и такие решения нужно принимать широким сообществом *KHTML*, а не командой KDE. Также требуется переписать несколько основных аспектов старого *KHTML*, включая виджеты форм, апплеты Java, код подключаемых модулей Netscape и общую интеграцию с KDE.

Проект Decibel

Еще один проект, способный принести заметные изменения в часть приложений – это Decibel, новый KDE API для мгновенных сообщений и видеоконференций. Может показаться, что KDE плодит проекты, не задумываясь о завтрашнем дне – особенно если учесть, что Phonon задался подобной целью встраивания функциональности VoIP в API. Но Decibel предназначен служить основой для коммуникаций в реальном времени, а не просто добавлять функциональность VoIP. И он будет прозрачным. Его идея в том, что вам не нужно знать, какой IM-протокол используют те, с кем вы общаетесь: Decibel просто сообщит вашим приложениям, когда станет возможна беседа, а все остальное – за сценой.

Приятно, что Decibel не собирается игнорировать другие проекты-аналоги. Уже существуют два хорошо определенных базиса для коммуникаций. Первый – Telepathy, ставший стандартом, по которому оцениваются другие. Он гнездится на http://freedesktop.org и использует D-BUS для предоставления соединения в реальном времени общим сервисам, типа IRC, мгновенных сообщений, голоса и видео. Чтобы реализовать все по максимуму, он не чурается и других стандартов, например, протокола Jabber, но Telepathy тяготеет к Gnome. Многие годы конкурирующей спецификацией была Tapioca, реализованная с использованием библиотеки *Qt*. По счастью, у программистов Tapioca возобладал здравый смысл: они сделали свою спецификацию совместимой с Telepathy – и теперь гармонично сосуществуют. Decibel будет использовать привязки *Qt* для Tapioca. Может показаться, что команда разработчиков KDE создает собственную реализацию *Qt*-механизма ради него. Но те дни, мы надеемся, прошли. В *Qt* 4 объединяются многие изменения, нужные проекту KDE, так что мы не должны бы снова увидеть виртуально идентичные классы KIconView и QIconView.

В Decibel реализована штука под названием Housten, это слой взаимодействия с Telepathy. Он включает Account Manager, Protocol Manager и Component Manager. Account Manager обрабатывает информацию о возможности соединения, например, находится ли пользователь в сети или отключен, а Protocol Manager определяет, каким образом будет подключаться каждый пользователь. Соединение Jabber – часть комплекса Telepathy, которая должна привлечь пользователей. Component Manager используется для управления данными приложений.

Phonon, *KHTML* и Decibel намереваются изменить отношение среднего пользователя к настольным Linux-системам. Мультимедиа полностью интегрируется на уровне рабочего стола, web-страницы будут отображаться быстрее, чем с *Firefox*, и с той же совместимостью, как у *Safari*. И болтовня с друзьями станет так же привычна за рабочим столом, как управление файлами. Команда разработчиков KDE 4 воспользовалась этим, чтобы рассмотреть, для чего пользователи сейчас применяют настольные системы и какие происходят изменения.

»

Что нам готовит Kopete





- » Вертикальные вкладки в окне чата
- » Возможность создавать глобальные персональные данные
- » kopete:/ KIO slave как полностью виртуальная файловая система
- » Способность сохранять все контакты в адресной книге KDE

Мэтт Роджерс [Matt Rogers], лидер проекта *Kopete*: «Decibel может стать крайне важной частью *Kopete* в KDE 4, и есть люди, которые думают над тем, как сработаются Decibel и *Kopete*. С нашей стороны, я могу только радоваться этому. Правда, мы еще не знаем, чему радоваться, потому что первые фрагменты кода Decibel только-только [в середине ноября] перешли в Subversion-репозиторий KDE.

Команда разработчиков *Kopete* оказывает некоторое влияние на разработку и спецификации Decibel, хотя большинство работ происходит в сфере проектов Telepathy и Tapioca. Несколько разработчиков Telepathy в этом году были на Akademy, и мне говорили, что от их присутствия было немало пользы».



› Улучшения движка KHTML поставят *Konqueror* в один ряд с Apple *Safari* – больше глаз, меньше ошибок.

KDE 4: хакнем все

Даже те, кто не умеет программировать, поймут по приведенному ниже фрагменту кода, насколько просто создать аудиоплейер в ваших приложениях. Всего в пяти строках мы закодировали возможность воспроизводить OGG-файлы.

```
AudioPlayer *player = new
        AudioPlayer (Phonon::MusicCategory, this);
player->play(KUrl("file://tune.ogg"));
player->seek(milliseconds);
player->pause(); player->stop();
```

Этот подход применяется многими новыми API в KDE, и он означает, что каждому под силу обогатить любое приложение рабочего стола.





Получите свежий KDE



Живите на передовой – ищите ошибки, давайте советы и стройте будущее.

Чтобы испытать последний релиз KDE 4, достаточно установить последние пакеты Kubuntu. К сожалению, вы не найдете больших изменений по сравнению с KDE 3; многие из основных приложений сейчас являются частью релиза, но ни одно из них не готово использовать преимущества новых API.

Пройдет еще месяцев шесть, пока вы ощутите реальные изменения, и чтобы быть способным их разглядеть, нужно постоянно обновлять вашу версию KDE. Здесь описываются шаги, необходимые для получения самой свежей версии, независимо от того, являетесь ли вы разработчиком KDE или энтузиастом-любителем. Мы использовали Ubuntu, но наши инструкции должны работать и на других дистрибутивах, поскольку здесь нет специфических требований. Новичкам на заметку: при разработке KDE 4 используется система контроля версий *Subversion*. Она отслеживает все изменения, сделанные разработчиком, а также управляет сотнями разработчиков, вносящих изменения в один и тот же проект. Не позволяйте ей вас запугать, если вы не умеете писать код: последнюю версию может установить каждый. Сейчас мы расскажем, как это сделать.

> «Многие проекты взывают к новым разработчикам, дизайнерам и художникам.»

1 Создайте нового пользователя

Чтобы надежно изолировать KDE 4 от остальной системы, лучше установить KDE 4 под новой учетной записью, созданной специально для целей запуска приложений KDE 4. Это можно сделать либо в менеджере учетных записей вашего дистрибутива, либо из терминала, набрав **su -c 'adduser kdefour'**. Вам нужно создать две папки в домашнем каталоге нового пользователя: одну с именем **Qt-copy**, а вторую – **kde4**. Также вам понадобится добавить некоторые переменные окружения, чтобы сохранить установку KDE 4 изолированной в вашем домашнем каталоге. Добавьте следующее в конец файла **.bashrc**, принадлежащего новому пользователю:

```
export REPOSITORY=svn://anonsvn.kde.org/home/kde
export QTDIR=~/qt-copy
export KDEDIR=~/kde4
export DBUSDIR=~/kde4
export KDEDIRS=$KDEDIR
export PATH=$QTDIR/bin:$KDEDIR/bin:$DBUSDIR/bin:$PATH
export LD_LIBRARY_PATH=$DBUSDIR/lib:$QTDIR/
lib:$KDEDIR/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=$KDEDIR/lib/kde4/plugins/
export PKG_CONFIG_PATH=$QTDIR/lib:$DBUSDIR/lib/
pkgconfig
export QTEST_COLORED=1
export XDG_DATA_DIRS=$KDEDIR/share
export XDG_CONFIG_DIRS=$KDEDIR/etc/xdg
```

2 Установите новую систему сборки, CMake

CMake заменяет старые пакеты Automake, на которые разработчики KDE полагались ранее. *CMake* теперь обрабатывает все сценарии компиляции для приложений, и должна намного упростить обработку зависимостей сборки. Инсталляция *CMake* проста: скачивание последнего пакета с www.cmake.org, распаковка полученного файла, затем переход во вновь созданный каталог и ввод следующих команд, для установки *CMake* в локальный каталог **kde4**:

```
./configure --prefix=~/kde4
make; sudo make install
```

3 Добавьте обмен сообщениями D-BUS

KDE 4 использует D-BUS для взаимодействия между процессами. D-BUS – эффективная и кросс-платформенная замена для старого DCOP. Найти ее можно на влиятельном сайте Freedesktop.org, и по умолчанию она устанавливается со многими последними дистрибутивами. Если вы работаете на старом дистрибутиве, вам может потребоваться скомпилировать и установить собственную версию. Обычно это заключается в скачивании последнего основного релиза с http://dbus.freedesktop.org, распаковке пакета и вводе следующих команд:

```
./configure --disable-qt --disable-qt3 --prefix=$DBUSDIR
make; sudo make install
```

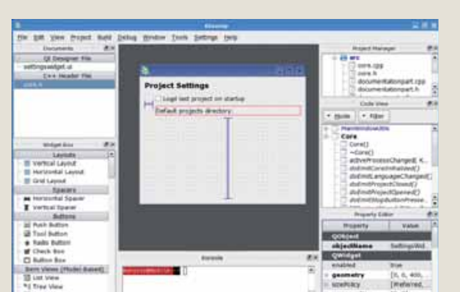
4 Получите свежую версию Qt

Прежде чем погрязнуть в исходном коде KDE, вам нужно скачать и установить последнюю версию инструментария *Qt*, от которой KDE наследует все свои внутренние прелести. *Subversion*-репозиторий KDE содержит версию, на которой работают разработчики KDE, и ее можно скачать и установить тем же способом, что и сам KDE:

```
svn co $REPOSITORY/trunk/qt-copy
cd qt-copy; ./apply_patches
```

1

Что нам готовит KDevelop



» Модификация большей части кода
» Поддержка большего числа языков
» Новый интерфейс сборки Cmake

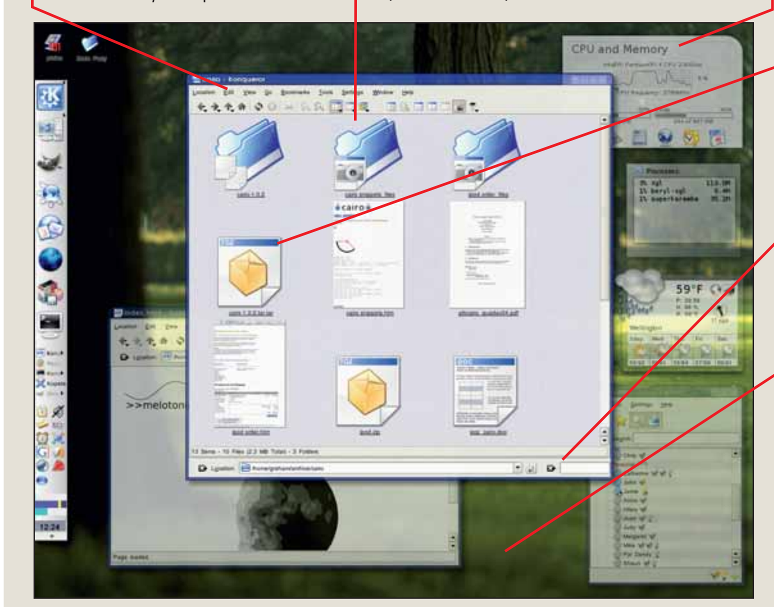
Мэтт Роджерс [Matt Rogers], ведущий «хранитель» KDevelop: «Было очень важно выкинуть всякий хлам из *KDevelop 4*, поскольку в *KDevelop 3* было слишком много неподдерживаемого кода, который мало кого волновал. Не думаю, что «глубокая переработка» неизбежна, когда выполняется крупное обновление *Qt* или KDE – все зависит от предпочтений людей, работающих над проектом. Мы повторно используем в *KDevelop 4* немало кода и идей из *KDevelop 3*. Например, нынешний менеджер проектов в *KDevelop 4* – прямой потомок кода, разработанного для *KDevelop 3*. Поддержка GNU *autotools* и *Qmake* в *KDevelop 4* также будет развитием фрагментов, уже присутствующих в *KDevelop 3*. Конечно, мы кое-что перепишем с учетом новых технологий в *Qt 4*, но это отнюдь не переработка до основания – на это ушла бы целая вечность! Я думаю, настройка приложения станет проще, так как все больше и больше разработчиков учитывают мнение людей, занимающихся эргономикой, которые предоставляют свои знания через различные проекты, типа OpenUsability, и список рассылки kde-usability.»

Как может выглядеть KDE 4: мнение нашего художника

Оформление окна
Новый оконный менеджер будет снабжен миловидным *Compiz* и вернет пользователям виртуальный пейджер рабочих столов, который сейчас с *Compiz* не работает.

Konqueror
Файловый менеджер KDE получит «подтяжку лица», чтобы упростить интерфейс пользователя без потери гибкости. Капитально отремонтированный *KHTML* поможет web-страницам отображаться более точно.

Элементы интерфейса
Виджеты в стиле *SuperKaramba* станут «сливками общества» под KDE 4, предоставляя вам полный контроль над выбором дополнительных функций.



Стандартизированные иконки
Иконки будут трансформированы из растра в векторный формат SVG. Это означает, что вы получите современный лоск на своем рабочем столе, независимо от разрешения экрана.

Поиск по всей системе
Всепроникающий «настольный» поиск позволит отслеживать ваши данные из любого приложения, не только по имени файла, но и по содержимому.

Дополнительная скорость
Обновление до *Qt 4* означает, что KDE 4 будет использовать меньше памяти, чем предыдущий релиз. *Qt* также ускорит работу KDE и упростит портирование приложений KDE на другие ОС.

```
./configure -qt-gif -no-exceptions -debug -fast -prefix $QTDIR -confirm-license
make
```

5 Получите KDE 4!

Теперь, когда у нас есть все основные зависимости для KDE, можно скачать и собрать рабочую копию KDE 4. Это нужно делать за три шага, поскольку приложения KDE требуют, чтобы сначала были собраны и установлены библиотеки и базовые файлы. Общая процедура такова:

```
svn co $REPOSITORY/trunk/KDE/kdelibs
mkdir kdelibs/build; cd kdelibs/build
cmake -DCMAKE_INSTALL_PREFIX=$KDEDIR ..
make -k; make install
```

Для установки базовых файлов замените **kdelibs** на **kdebase** в приведенном выше коде. Как только сборка и установка завершатся, все необходимые для запуска приложений KDE 4 части будут на месте. Их можно найти в каталоге **kde4/bin**, а **kdebase** включает предметы первой необходимости, например, текстовый редактор *Kate* и web-браузер *Konqueror*. Чтобы вызвать какое-нибудь приложение, сначала убедитесь, что D-BUS работает, затем запускайте двоичный файл. Чтобы запустить *Kate*, используйте следующие команды:

```
eval 'dbus-launch --auto-syntax'
~/kde4/bin/kate
```

6 Держитесь в курсе

Вы можете просматривать репозиторий KDE 4, заходя на http://websvn.kde.org/trunk/KDE. Здесь можно оценить прогресс, достигнутый в других основных проектах KDE, и скачать их, используя ту же самую процедуру, что и в Шаге 5. Просто замените **kdelibs** именем проекта, представленным на сайте WebSVN, затем скачайте, откомпилируйте и установите. В случае с базовой инсталляцией KDE 4 сравнительно просто отслеживать прогресс разработки. *Subversion* имеет команду для обновления исходного кода до последней версии, размещенной на сервере: вам просто нужно войти в каждый из основных подкаталогов, которые вы скачали, и запустить:

```
cd kdelibs
svn update
cd build
make; sudo make install
```

Мы использовали **kdelibs** в примере выше, потому что его по-прежнему нужно обновлять и устанавливать первым, затем – **kdebase**, чтобы не нарушить баланс зависимостей, необходимый для сборки и установки других пакетов.

Отсюда и в вечность

В ближайшие шесть месяцев KDE 4, скорее всего, не выйдет, но разработка идет полным ходом. Пока вы это читаете, разработчики создают новые функции. В результате – сейчас самое время для того, чтобы присоединиться к ним. *Qt 4* намного понятнее, чем предыдущие версии, и многие проекты взывают о дополнительных разработчиках, дизайнерах интерфейса и художниках. А вследствие упора на удобство использования возникает и требование в улучшении документации. Если вы всегда хотели посодействовать сообществу KDE, KDE 4 – прекрасная возможность.

Первое, что вам нужно сделать, – это подписаться на списки рассылки KDE, как пользовательский, так и для разработчиков, затем связаться с проектом, который вас заинтересует. Наконец, не забывайте писать нам о том, как идут дела, на letters@linuxformat.ru. LXF

- » www.kde.org/mailinglists – Списки рассылки
- » http://ev.kde.org – Некоммерческая организация при KDE
- » http://dot.kde.org – Для новостей разработки KDE
- » http://appeal.kde.org – Дополнительная информация об Appeal
- » http://plasma.kde.org – Домашняя страница Plasma
- » http://solid.kde.org – Solid
- » http://phonon.kde.org – Phonon
- » http://decibel.kde.org – Decibel

Что за штука… Микроформаты?

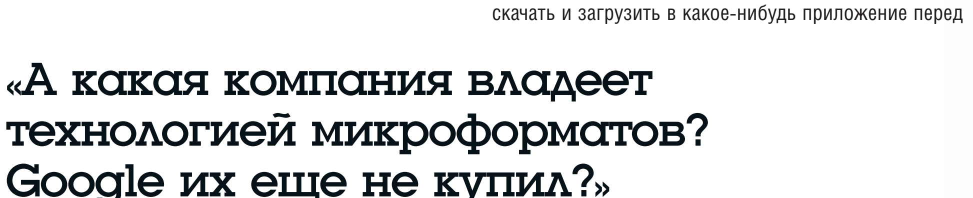
Хотите ли вы сделать ваш HTML еще более информативным? Ну конечно, да! **Брайан Суда** сообщит вам кое-что о новых форматах данных, позволяющих обогатить ваши web-сайты.

» Громкие термины Web 2.0 постепенно приелись, и сейчас я все чаще слышу о каких-то микроформатах. Не могли бы вы объяснить мне, что это такое?

Микроформаты придают дополнительный смысл простым и широко распространенным в Интернете типам данных. Например, их можно использовать, чтобы структурировать контактную информацию или календарные события. Хотя термин «микроформаты» рассматривают как один из модных «кирпичиков» Web 2.0, сама идея включения дополнительных смысловых элементов в HTML витает в воздухе с 1997 года.

» Это что, новый язык программирования, который мне придется изучать? Вообще-то мне уже хватает C# и Tcl.

Нет, это не новый язык. Микроформаты понятны для человека и работают внутри HTML 4.01 или XHTML. Они базируются на имеющихся стандартах, так что все, что вам понадобится знать – это HTML плюс несколько дополнительных атрибутов элементов.


«А какая компания владеет технологией микроформатов? Google их еще не купил?»

» Ага, так значит микроформаты – это просто разновидность CSS?

Вовсе нет, просто микроформаты, как и CSS, используют атрибут class. На этом их сходство заканчивается. CSS – это набор правил, указывающих, в каком *стиле* следует представить данные. Микроформаты же предназначены для того, чтобы добавить новый смысл к уже существующему тексту. Атрибут class был придуман W3C для того, чтобы для новые идеи (такие как микроформаты) могли хранить свои метаданные в рамках HTML.



» Хорошо, и как же мне приступить к добавлению микроформатов в HTML?

Вообще говоря, дополнительные значения можно добавлять к HTML-документу всего в трех местах – в тэг meta, в атрибуты class и rev/rel. Поскольку микроформаты предназначаются для видимого пользователю содержимого страницы, они, в основном, фокусируются на двух последних – атрибутах class и rev/rel.

» А расскажите немножко про историю разработки микроформатов!

История развития микроформатов началась задолго до того, как было придумано это слово. В 2002 году несколько энтузиастов, Тантек Сейлик [Tantek Celik], Мэтью Малленвег [Matthew Mullenweg] и Эрик Мейер [Eric Meyer] начали работу над XHTML Friends Network (XFN). Большую часть их работы сейчас можно увидеть на сайте Global Multimedia Protocols Group (http://gmpg.org). Их целью было «соединять людей просто и последовательно». Эта идея простого постепенного наращивания семантики впоследствии и легла в основу микроформатов.

» Почему так важно, чтобы микроформаты были понятны человеку?

Человеко-читаемость очень важна по нескольким причинам. С глаз долой – из сердца вон. То, что вы видите каждый день в окне браузера, скорее будет поддерживаться в актуальном состоянии, чем какой-нибудь закодированный файл на сервере, который нужно скачать и загрузить в какое-нибудь приложение перед тем, как посмотреть. Если данные хорошо видны, то они будут более свежими и полезными для всех.

» Да, но все мои данные хранятся в CMS-системе, которая заботится об их актуальности. Зачем мне могут понадобиться микроформаты?

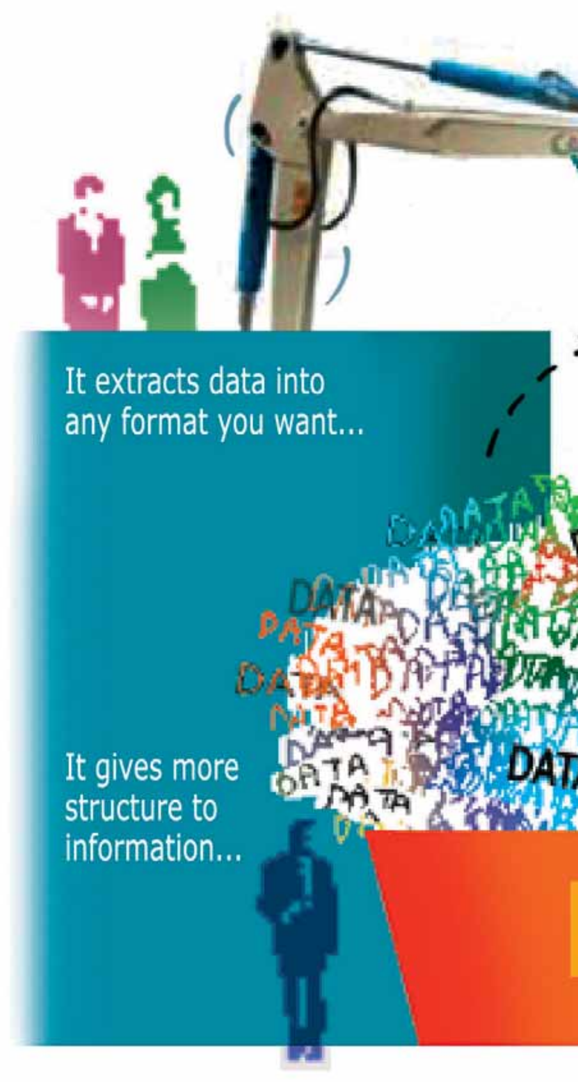
Действительно, многие люди используют XML или базы данных, чтобы регулярно обновлять свой web-сайт. Это решает проблему поддержания файлов, отличных от HTML (таких как RSS, vCards и iCalendar) в актуальном состоянии, но это никак не увеличивает информативность ваших HTML-страниц. Обыкновенный посетитель вашего сайта наверняка не имеет доступа к вашей CMS-системе, по крайней мере, если вы не предоставляете ему какой-нибудь API. А при использовании микроформатов HTML-страница сама становится таким API.

» Мой сайт не делает ничего особенного. Я не понимаю, зачем бы ему мог пригодиться API.

Представьте себе, что на вашем сайте опубликованы данные о сотрудниках некой компании в виде XML-файлов, а мое приложение понимает только формат vCard. Мне придется искать специальную программу, чтобы выполнить преобразование между форматами. Понимаете? Если не размещать данные прямо в HTML, вашей CMS-системе придется поддерживать 30 разных форматов, чтобы удовлетворить всех клиентов! А при использовании микроформатов становится возможным извлекать информацию прямо из HTML и преобразовывать ее к любому виду – даже к такому, о котором вы и не думали.

» Значит, существуют разные форматы «микроформатов»?

Конечно! Когда микроформаты начинали свое существование, их создатели тщательно изучили, что публи-



куется в сети чаще всего. В число наиболее популярных тем вошла информация о людях, местах и событиях. Поэтому среди популярных микроформатов оказались hCard, повторяющий функциональность vCard и описывающий людей, места и организации, а также hCalendar, похожий на iCalendar и описывающий различные события. Другие микроформаты перекрывают возможности резюме (формат «hResume»), обзоров (можно описывать фильмы, продукты, сайты), цитатников, синдикаторов и так далее.

RSS и Atom – это микроформаты?
Нет, это старые добрые XML-форматы. Существует специальный микроформат, hAtom, который можно использовать для того, чтобы преобразовать в RSS или Atom обычную HTML-страницу. Микроформаты позволяют вносить новую семантику внутрь простого HTML.

Хорошо, тогда посмотрите на все эти сайты, публикующие устаревшие варианты RSS – вы хотите сказать, что если они разметят свои HTML-страницы при помощи микроформатов, то я смогу конвертировать их в Atom 1.0?
Точно, вы поняли идею! Представьте, что владелец сайта заключил разовый контракт с разработчиками, чтобы они создали сайт и сделали для него RSS-ленту. Пусть в тот момент последней версией Atom была 0.3, и поэтому разработчики использовали именно ее. Сейчас Atom уже дорос до 1.0, но все контракты давно закончились, а ресурсов на обновление системы синдикации в бюджете нет, и поэтому сайт застыл на уровне 0.3.

Микроформаты могут помочь предотвратить такую проблему. Дополнительная разметка, которой они снабжают HTML, позволяет получить информацию в любом формате, не ограничиваясь имеющимся Atom 0.3

Ага, мы все это время говорим о добавлении новой семантики. Я вот тут все жду возможности выложить в Интернет свою коллекцию марок, как бы мне это сделать?
Одной из ключевых особенностей микроформатов является то, что они не обладают бесконечной гибкостью. Решение всех мировых проблем – не их предназначение. Они созданы для решения конкретных задач, существующих в современном Интернете, и структурирования имеющейся информации. В идеале микроформаты должны позволять достичь максимальных результатов с минимальными затратами. Они основываются на том, что на сайтах уже опубликованы гигабайты и гигабайты данных, и нет нужды изобретать велосипед – достаточно немножко структурировать то, что уже есть, чтобы с ним было удобнее взаимодействовать.

А что, кто-то использует микроформаты в реальности?
О, многие сайты делают это. При помощи минимальной корректировки HTML-шаблонов многие сайты могут получить тысячи, а то и миллионы элементов микроформата всего за несколько минут. Такие сайты, как Yahoo Local, Yahoo Tech, Flickr, Meet-up, Upcoming, Eventful, Edgeio, Technorati, Ice Rocket, .Mac mail on Apple используют различные микроформаты.

Это все звучит просто замечательно, но какая компания владеет технологией микроформатов? Google их еще не купил?
Микроформаты не принадлежат никому. Прежде чем новый микроформат вступит в силу, он должен быть одобрен большой группой добровольцев и всемирным сообществом. Они же занимаются поддержкой имеющихся стандартов и документированием (все разработки предоставляются под очень либеральными лицензиями). Поскольку микроформаты не принадлежат ни компании, ни частному лицу, их использование нельзя ограничить решением Совета директоров. Это открытое сообщество, разрабатывающее открытые форматы данных, которые делают Интернет еще более приятным место для общения.

Да, я должен признать, это звучит очень вдохновляюще! Где я могу почитать про микроформаты?
Официальный сайт расположен по адресу http://microformats.org. Там вы найдете wiki, список рассылки, блог, ссылки на IRC-каналы и так далее. Кроме него, существует несколько отдельных сайтов, посвященных микроформатам – http://microformatique.com, www.whymicroformats.com. Вы также можете посмотреть списки ресурсов про микроформаты в различных каталогах – http://del.icio.us/tag/microformats, http://technorati.com/tags/microformats и http://ma.gnolia.com/tags/microformats. Даже новые Live Clipboard (http://snipurl.com/10so0) и Live Writer (http://windowslivewriter.spaces.live.com) от Microsoft используют микроформаты. LXF





www.ellywalton-illustrations.com

Tcl Нестандартное программирование

LINUX FORMAT на DVD
» Tcl 8.4.13
» Moodss 21.3
» TkDVD 4.0.3
» TkDiff 4.1.3

Хотите взлететь повыше, чем C и Perl? **Майк Сондерс** открывает новую серию публикаций о менее известных, но не менее интересных языках программирования...

Вы строчили на C, баловались с Basic и программировали на Perl. А может, формулировали задачи на Фортране, применяли Python и даже атаковали Ada. Но ваш аппетит к кодированию не утолен… К счастью, в мире программирования всегда есть пища для пытливого ума. Изучение новых языков – прекрасный способ расширить профессиональные знания. Да и резюме, включающее не только приевшийся C, произведет большее впечатление на работодателя.

Мы начинаем серию статей, посвященных не совсем обычным языкам программирования. Они далеки от основного русла [mainstream], но тоже служат важным целям, а изучать их – одно удовольствие. Мы предполагаем, что вы уже владеете базовыми понятиями, поэтому не будем тратить время, объясняя, что такое переменная или цикл. Опустим также некоторые чисто технические подробности: их можно найти в документации.

В данной серии мы выясним, откуда взялись эти языки и что делает их интересными сегодня, и рассмотрим ряд практических примеров, которые позволят вам продолжить освоение языка самостоятельно. Правда, на четырех страницах особо развесистое приложение не проанализируешь, но трамплин для прыжка к написанию собственных программ мы вам подставим. И если у вас получится что-то действительно стоящее, мы сможем опубликовать это на нашем *DVD*!

Тиклни меня

Наша серия начнется с рассказа о Tcl (часто произносится как английский глагол tickle – «щекотать»). Этот интерпретируемый язык придуман в 1988 г. на родине BSD Unix и других пионерских разработок в компьютерных технологиях – в Калифорнийском университете Беркли. Его основоположник, Джон Остерхаут [John Ousterhout], создал собственный «командный язык инструментов» – Tool Control Language, насмотревшись на недопеченные встраиваемые языки от других разработчиков. Его основными целями были возможность простого и эффективного встраивания в более крупные приложения и ускорение процесса прототипирования – создания тестовых приложений, служащих для оценки какой-либо идеи.

Сейчас эти задачи успешно решаются и другими языками (например, Python), и невольно возникает вопрос: а зачем изучать Tcl – что в нем особенного? Во-первых, он облегчает разработку программ с графическим интерфейсом. Для работы с *PyGtk* (реализация Python на *GTK*) необходимо изучить структуру и особенности C-ориентированного инструментария, а Tcl использует *Tk*, несложный, но гибкий набор элементов GUI, существенно упрощающий организацию пользовательского интерфейса. Мы рассмотрим *Tk* более подробно чуть ниже.

Многие козыри языка Tcl уже стянули Perl, Python и PHP, но Tcl по-прежнему используется широко: именно его применяли при разработке программного обеспечения модуля оператора буровой вышки компании Shell и космического телескопа Хаббла. Это очень серьезная работа, так что в зрелости и стабильности Tcl сомневаться не приходится. Благодаря простоте освоения и интеграции в другие приложения язык принят на вооружение фирмами Oracle и IBM. В списке приложений с открытым исходным кодом, приведенном на сайте http://freshmeat.net/, более 300 Tcl-проектов; среди них программа для мониторинга серверных процессов *Moodss*, программа для записи дисков *TkDVD* и помощник кодировщика *TkDiff*.

Интерпретатор языка Tcl входит в состав большинства дистрибутивов и обычно устанавливается по умолчанию. Если вашем дистрибутиве его почему-либо нет, возьмите Tcl на нашем *DVD* в разделе **Разработка**. Tcl – интерпретируемый язык, и вы обойдетесь без текстового редактора и командной строки, обвешанной ключами-флагами: все будет ясно с первого взгляда. Для запуска интерпретатора Tcl введите **tclsh** в командной строке. В строке приглашения появится подсказка Tcl – лаконичное %; основы языка можно исследовать прямо с ходу. С чего начать? Конечно, с программы, выводящей на экран строку «Hello, World!»

```
puts "Hello, world!";
```

Здесь puts – команда put string, которая выводит заданную строку в стандартный поток вывода stdout. Можно записывать несколько операторов в одной строке, разделяя их точкой с запятой, или размещать каждый оператор на новой строке – это дело вкуса.

После вывода попробуем ввод:

```
gets stdin foostring;
```

Это запрос на ввод строки; результат сохраняется в переменной foostring. Обратите внимание на важную особенность языка: переменные не нужно объявлять до их использования. Чтобы вывести результат, воспользуемся уже известной нам командой puts:

```
puts stdout $foostring;
```

Ввод значений числовых величин также не представляет особых сложностей. Рассмотрим пример:

```
gets stdin x;
gets stdin y;
expr $x*$y;
```

Первые две строки попросят вас ввести числа; так и сделайте. Введенные числа сохранятся в виде строк в переменных x и y, затем команда expr вычислит заданное ей выражение произведения x*y и отобразит результат. Просто и мило! Переменные Tcl – не статически типизированные, и их можно определить в любое время. Значения присваиваются переменным командой set:

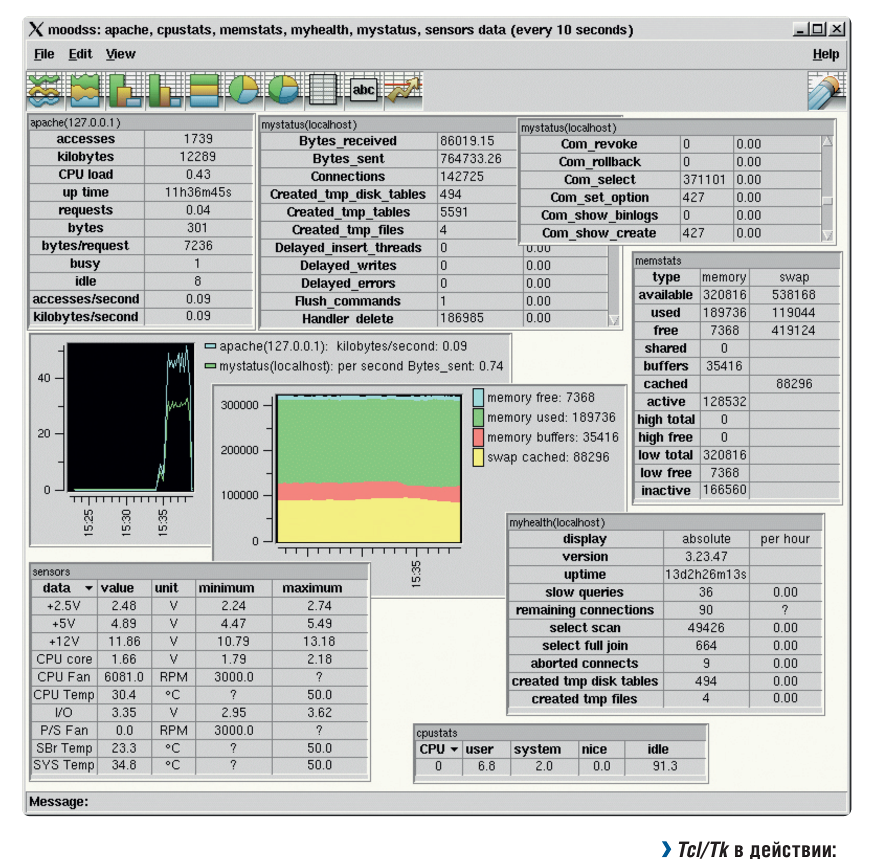
```
set x 7;
echo $x;
gets stdin y;
set x $y;
echo $x;
```

Переменной x присваивается значение 7, x выводится на экран, затем вводится значение y, копируется в x, и x выводится снова.

Перейдем к операторам условия. Наберите коды следующих примеров в текстовом редакторе, сохраните в файлах, а затем запускайте интерпретатор командой tclsh <имя_файла>. Рассмотрим оператор условия if/else:

```
gets stdin x;
if {$x>10} {
puts "Bigger than ten";
} else {
puts "Less than ten";
}
```

Правда, похоже на C? Оператор цикла while работает в той же манере:



› ***Tcl/Tk* в действии: *Moodss* анализирует *Apache*.**

```
gets stdin x;
while {$x<=10} {
puts "Enter a number bigger than 10";
gets stdin x; # User inputs number
}
```

Пока вы не введете число, большее 10, оператор будет требовать ввода снова и снова. В строке с gets символ # означает начало комментария, а строка перед ним в Tcl должна заканчиваться точкой с запятой. В коде на Tcl вы нередко встретите «вложенные команды» ('nested commands'), заключенные в квадратные скобки. Их результат может присваиваться другим переменным, например:

```
set mystring "Hello, world!";
set x [string length $mystring];
puts $x;
```

С помощью вложенной команды string length в переменную x записывается длина строки mystring.

«Tcl использовался в ПО телескопа «Хаббл»: это серьезная работа, требующая от языка надежности и зрелости.»



Процедуры

Итак, мы освоили ввод/вывод, переменные, операторы условия, цикла и комментарии. Теперь поговорим о процедурах. Для объявления процедуры используется команда proc, после чего процедура становится просто командой Tcl. Процедура должна быть объявлена до ее использования – иначе ждите сообщения об ошибке. Пример простой процедуры, перемножающей два числа, приведен ниже:

```
proc multiply {x y} {
set z [expr $x*$y];
return $z;
}
puts [multiply 3 4];
```

В первой строке скрипта объявляется процедура multiply и приводится список ее аргументов (x и y), заключенный в фигурные скобки. »



» Ниже в фигурных скобках определяется тело процедуры, а ее результат передается в основную программу оператором return. *Tclsh* первым делом выполнит последнюю строку скрипта (она находится вне процедуры), а вложенная команда вызывает процедуру, с тем, чтобы возвращаемое ею значение вывелось на экран командой puts. Запустив код, вы увидите 12.

В данном примере оператор return $z; можно опустить: Tcl возвращает значение последнего оператора процедуры как результат. Но лучше указывать оператор return явно: читать ваш код будет значительно легче, особенно тем, кто привык программировать на C.

Приведенный пример также демонстрирует область действия переменной. Аргументы и переменные, определенные в теле процедуры, являются локальными, и их нельзя использовать за пределами процедуры. Например, если вы попробуете вывести значение переменной z в основной программе (puts $z;), то получите сообщение об ошибке «переменная не определена» ('no such variable'). Чтобы переменная была доступна в любом месте программы, ее необходимо объявить глобальной с помощью команды global:

```
global z;
```

Et voila! Теперь к переменной z можно обращаться и вне процедуры.

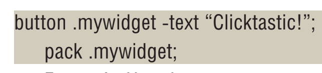
Разработка программ с GUI

Итак, основы Tcl изучены, и теперь вы сами можете попробовать что-нибудь написать. Список полезных сайтов, где можно найти дополнительные материалы, приведен на стр. 41.

А сейчас мы займемся по-настоящему увлекательным делом – разработкой программ с графическим интерфейсом. Как упоминалось выше, к Tcl подвязан *Tk* – готовый набор виджетов (так называют элементы пользовательского интерфейса: кнопки, переключатели и т.д.), поэтому подобные программы занимают всего несколько строк кода. Если вы уже писали программы с использованием *Qt* или *GTK*, то оцените невероятную скорость разработки в Tcl/*Tk*: больше не нужно трудиться над кодом инициализации и долгими часами старательно изучать API.

> «Tk использует только базовые библиотеки X, поэтому быстр и нетребователен к памяти.»

Если консольные программы на языке Tcl используют интерпретатор *tclsh*, то приложения с графическим интерфейсом Tcl/*Tk* используют *wish* ('windowing shell'). Этот интерпретатор выдает простейшую форму, на которой можно размещать виджеты вашего будущего приложения. Сейчас вы увидите, как все просто! Запустив *wish*, введите следующие команды:



```
button .mywidget -text "Clicktastic!";
pack .mywidget;
```

Готово! На форме появилась кнопка с надписью **Clicktastic!**. Ее даже можно нажать, но ничего не произойдет, потому что действие кнопки мы еще не определили. Команда button в первой строке создает кнопку с именем .mywidget (имена виджетов должны начинаться с точки). Используя это имя, мы сможем ссылаться на данный объект в дальнейшем. Опция -text задает надпись на кнопке.

Итак, создан объект .mywidget. Для отображения виджета в окне *wish* мы должны «упаковать» его в это окно (командой pack). «Упаковка» (packing) – это система размещения и упорядочивания виджетов в окне программы или родительском виджете (сейчас у нас только одна кнопка, и окно пристроилось под нее автоматически). Виджеты можно изменять «на лету» – введите в окне терминала такой код, в добавление к предыдущим двум строкам:

```
.mywidget configure -foreground green;
```

Текст **Clicktastic!** позеленел!

Небольшое отступление: по сравнению с *GTK* или *Qt* внешний вид виджетов *Tk* может показаться не совсем привычным. Дело в том, что *Tk* использует только базовые библиотеки X, а значит, не обеспечивает особых эффектов прорисовки (в частности, сглаживания); зато он быстр и нетребователен к памяти.

Пора заставить нашу кнопку что-то делать при нажатии. Снова запустите *wish* и введите в окне терминала следующие команды:

```
button .mywidget -text "Don't click me" -command { puts "I said no!" };
pack .mywidget
```

Опция -command команды button описывает действия, выполняемые при нажатии кнопки, в фигурных скобках: в данном случае это оператор puts, и он выводит указанную строку в стандартный поток вывода. Теперь, как только мы нажмем на кнопку, этот текст появится в окне терминала!

Диспетчер запуска приложений

Вы уже неплохо вооружены средствами для разработки графического интерфейса. В завершение темы, создадим небольшую, но полезную программу – диспетчер запуска приложений. На сайтах Freshmeat и SourceForge пасутся табуны проектов этого рода. Они особенно популярны для оконных менеджеров типа FluxBox, не имеющих своих кнопок быстрого запуска.

Применим полученные знания и напишем простенькое приложение для запуска ваших любимых программ (например, *Firefox* или *Thunderbird*) и управления системными сервисами (например, SSHD или Apache). Вот как будет выглядеть наш код:

```
#!/usr/bin/env wish
set sshcommand "/etc/init.d/ssh";
global sshcommand;
proc sshstart {} {
global sshcommand;

exec $sshcommand start &;
}
proc sshstop {} {
global sshcommand;
exec $sshcommand stop &;
}
proc launchbrowser {} { exec firefox &; }
proc launcheditor {} { exec emacs &; }
frame .app -borderwidth 10;
.app configure -background lightblue;
pack .app;
button .app.ssh-go -text "Start SSHD" -command sshstart;
button .app.ssh-end -text "Stop SSHD" -command sshstop;
button .app.browser -text "Web browser" -command
launchbrowser;
button .app.launcheditor -text "Text editor" -command
launcheditor;
.app.ssh-go configure -foreground green;
.app.ssh-end configure -foreground red;
pack .app.ssh-go .app.ssh-end .app.browser .app.launcheditor;
```

Наш менеджер предусматривает четыре кнопки: две для запуска и останова сервера SSH и еще две – для запуска *Firefox* и *Emacs*. Первая строка (вы могли встречать такую в скриптах *Bash*) указывает интерпретатор для обработки скрипта – *wish*, и благодаря ей уже не нужно набирать wish имя_файла, чтобы скрипт заработал. Сохраним скрипт

› **Результат нашей работы: красочное меню запуска программ!**



в файл, например, **/usr/bin/megatest** и сделаем файл исполняемым (**chmod +x /usr/bin/megatest**). Теперь мы можем запускать его командной строкой (**megatest**) или создать соответствующий ярлык на рабочем столе.

Следующие две строки определяют глобальную переменную sshcommand: это строка, содержащая команду запуска сервера SSH, она сэкономит вам набор команды. Кроме того, если SSH в вашей системе расположен в другом каталоге, достаточно будет отредактировать это определение, и не выискивать обращения к SSH по всей программе – совсем как с #define в С.

Далее идут четыре процедуры. Две последние уместились в одну строчку; тем больше места останется для статьи, а кстати и код становится компактнее. Процедуры sshstart и sshstop предназначены для запуска и останова сервера SSH и используют нашу глобальную переменную sshcommand: она служит параметром команде exec, которая, таким образом, обращается к серверу SSH, прибавив вторым параметром start либо stop. Учтите, что ваша программа потребует привилегий суперпользователя (root): только ему дозволено распоряжаться SSH!

Процедуры launchbrowser и launcheditor запускают соответственно *Firefox* и *Emacs*; они запустят все что угодно в пределах вашего $PATH, так что можете заменить и браузер, и редактор на свои любимые. Первый оператор головной программы создает виджет-рамку по имени .app, он будет служить контейнером для остальных виджетов. Рамки, как и кнопки, можно располагать в окне *wish*; .app позволит нам задать контур вдоль границы окна. Ширина контура назначается с помощью опции -borderwidth (здесь – 10 пикселей). Затем мы «упаковываем» созданную рамку в главное окно программы.

Далее с помощью знакомой команды button мы создаем объекты, соответствующие кнопкам быстрого запуска. Что интересно, для размещения кнопок внутри рамки мы должны использовать префикс .app в именах кнопок – тогда каждая из них будет «упакована» в родительский виджет (рамку). Очень мощный аппарат для создания сложных GUI!

Попробуйте изменить имя объекта .app.launcheditor на .launcheditor (не забудьте также изменить последнюю команду pack!). Запустите скрипт, и увидите, что кнопка запуска редактора Emacs оказалась вне рамки. Таким образом, составные имена объектов определяют их взаимосвязь.

С помощью команд configure задается цвет надписи на кнопках управления сервисом SSH – попробуйте изменить foreground на background и посмотрите, что получится. Наконец, мы «упаковываем» все виджеты, соответствующие кнопкам, в одну рамку. Готово! Теперь можете добавить другие кнопки и назначить им определенные действия по своему усмотрению.

Заключение

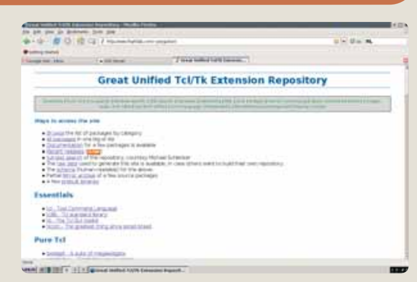
Итак, создание небольших графических приложений в Tcl/*Tk* – до изумления простая задача. Вот почему этот язык и его инструменты используются при создании ПО для объектов и систем, требующих надежной и безотказной работы и высокого быстродействия: ведь, к примеру, программе для управления телескопом нужен не элегантно сглаженный текст, а минимальный, четкий и легко модифицируемый код интерфейса. Добавьте к этому огромную библиотеку расширений, и получите язык без лишнего жира, зато с мускулами, способными справиться с решением множества задач.

Что дальше? Во врезке *«Ссылки»* приведены адреса сайтов с документацией и примерами скриптов – вы увидите, что можно приделать интерфейсы Tcl/*Tk* к инструментам командной строки. Кроме того, *Tk* поставляется с очень полезными примерами, обычно они находятся в каталоге **/usr/share/doc/Tk8.4/examples/** (цифры означают номер версии; подставьте вместо них вашу). Исходный код программ *Moodss*, *TkDVD* и *TkDiff* имеется на нашем *DVD*, из него можно почерпнуть немало идей для ваших проектов. Удачи вам! LXF

Расширения

Хотя в языке Tcl хватает команд для решения большинства задач, можно обогатить его функции с помощью расширений. Расширения Tcl аналогичны библиотекам для языка С или дополнительным модулям для Python. Самое известное расширение – это, конечно, Tk, но доступны и многие другие, обеспечивающие доступ к базам данных, обработку изображений и т.д.

Расширения Tcl обычно пишутся на С или С++, и для владеющих этими языками особых затруднений тут не будет. Хорошее руководство по созданию расширений для Tcl есть на www.equi4.com/pub/etc/extuse.html. См. также врезку *«Ссылки»* на этой странице.



› На сайте Gutter – огромный список расширений Tcl.

Ссылки

» Официальный сайт: www.tcl.tk
» Документация: http://wiki.tcl.tk
» Проекты: www.tcllinks.org
» Модули расширений: www.flightlab.com/~joe/gutter



» **Через месяц** Мы распотрошим Ruby, объектно-ориентированный язык – основу Ruby on Rails.

Новая серия! Свободной музыке – свободный инструментарий!

Трекеры



ЧАСТЬ 1: **Петр Семилетов** оторвался от Amiga и эмуляторов DOS, чтобы рассказать о программах-трекерах, имеющихся в Linux.

Этим материалом я начинаю небольшой цикл статей о создании и обработке музыки с помощью программных средств Linux. Не претендуя на полноту изложения материала, я расскажу о наиболее качественных и полезных (на мой взгляд) программах. Всевозможные кодеры и звуковые утилиты останутся «за бортом», потому что выходят за рамки заявленной темы. Я хочу рассказать именно о создании музыки в Linux.

Процесс этот, независимо от платформы, не подразумевает какой-либо устоявшийся, стандартный для всех набор программ и оборудования. Кто-то всё – от записи мелодий до сведения – делает на компьютере, а кто-то использует его только для сведения записываемых «вживую» инструментов – гитар, ударных и так далее. Часто применяется смешанный подход – микшируются как партии, сыгранные вживую, так и созданные с помощью различных программ – барабанных машин, виртуальных синтезаторов, MIDI-секвенсеров, которые тоже могут управлять виртуальными синтезаторами. MIDI-секвенсер (невесть почему у нас часто пишут еще и «секвенсор») – это, грубо говоря, программа, в которой вы можете нотами на нотном стане или квадратиками на временной шкале (так называемый «пианоролл») записать мелодию, которая будет воспроизводиться выбранным вами инструментом с помощью MIDI-синтезатора звуковой карты, внешним синтезатором, либо виртуальным синтезатором.

Исторически сложилось так, что сейчас для создания и обработки музыки в основном используется программное обеспечение для системы Windows XP. На втором месте идет Mac OS. Следом можно поставить Linux, но здесь возникает вопрос – как много музыкантов, а не любителей Linux, использует эту систему в качестве рабочей ОС?

Музыкальное программное обеспечение для Linux частично пытается повторить существующее для Windows и DOS, а частично воплощает в себе новые подходы – хороший пример тому звуковой сервер *Jack*, служащий как бы микшером, к которому подключаются другие звуковые программы, умеющие работать с *Jack*. Но если говорить непредвзято, то в Linux не существует программ класса *Steinberg Cubase SX/Nuendo*. Я бы назвал этот класс «студийным», потому что именно такое ПО используется на студиях звукозаписи. В Linux есть, конечно, «наш ответ Cubase» в виде *Muse* и *Rosengarden* – но это, пожалуй, «наш ответ» старым версиям *Cubase*, а не текущей. Это серьезные добротные программы, но другого калибра. О них мы поговорим в свое время, однако, начнем знакомство с музыкальным ПО для Linux с другой категории – трекеров.

Трекеры

В трекере можно писать музыку, не зная нот, не имея дорогой звуковой карты и прочего околозвукового оборудования. В само название заложена суть: трекер – «дорожечник». Музыкальная композиция трекерного формата называется модулем, и состоит она из паттернов – эдаких страниц, единиц композиции. Паттерн же, в свою очередь, содержит в себе представленные вертикально дорожки – каналы. Каждый канал поделен на строчки – ряды. А уж ряд содержит в себе такие данные, как ноту, инструмент или сэмпл (которым эта нота будет сыграна), а также громкость, панораму (расположение в стерео-пространстве) и эффект.

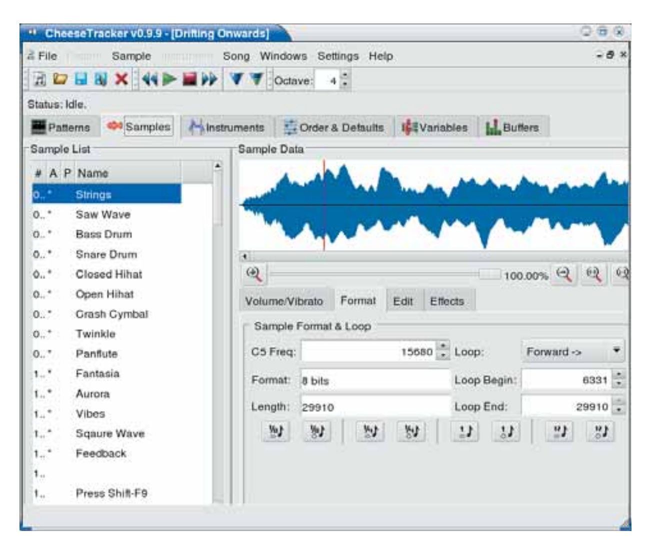
Как это выглядит на практике? Вы загружаете в трекер сэмплы или инструменты. Сэмпл – это, чаще всего, обыкновенный WAV-файл. Сэмплы продаются на дисках, их можно скачать в сети. Бывают сэмплы, записанные с настоящих живых инструментов, бывают – с «внешних» синтезаторов. Можно использовать сэмплы вокала, да чего угодно.

В трекерах также есть понятие «инструмента». Инструмент – это файл особого формата, в котором, для улучшения качества воспроизведения ноты, содержится несколько сэмплов. Допустим, вы хотите сыграть в трекере определенную ноту. При использовании для этого «одиночного» сэмпла, его основной тон (pitch) будет подогнан – сдвинут – чтобы соответствовать требуемой высоте звука. Как правило, делается это простым ускорением воспроизведения сэмпла. Надо ноту выше – сэмпл ускоряется. Надо ниже – замедляется. Само собой, это влияет на качество. В инструменте же может быть по сэмплу на каждую октаву, а можно вообще сделать инструмент, где будет по сэмплу на каждую ноту! Чем больше такая детализация, тем меньше трекеру приходится подгонять сэмплы под ноты, и звук становится более естественным. Такой подход применяется не только к трекерах, но, например, и в формате инструментов *Sound Fonts*, используемом в основном на звуковых картах Creative, которые обладают возможностью аппаратно загружать такие инструменты в память и воспроизводить.

Принято считать, что первый трекер появился в 1987 году – это была программа под названием *Soundtracker I*, и выпускалась она для платформы Amiga, хотя до этого были и трекеро-подобные программы для Commodore 64. После 87-го года на Amiga появились клоны Soundtracker, возникла целая субкультура музыкантов, использующих эти продукты для создания музыки. Затем трекеры перебрались в более современный (на тот момент) мир DOS, на платформу x86. Пожалуй, тогда был расцвет популярности трекеров и созданной с их помощью музыки. К концу девяностых годов, помимо других трекеров, существовало два мощнейших продукта этого разряда – *Fast Tracker 2* и *Impulse Tracker 2* (последний все еще доступен на www.lim.com.au/ImpulseTracker). Затем трекеры стали потихоньку перебираться в Windows. Появился трекер *ModPlug* и одноименная библиотека, которая вначале использовалась для воспроизведения трекерных модулей плейерами вроде *Winamp*, а нынче исправно играет ту же роль в плейерах для Linux.

Примерно в то время, когда эволюция Windows-трекеров завела их в область виртуальных синтезаторов (*Buzz*, *Psycle*), способных соперничать с лучшими виртуальными инструментами формата VST, Linux-трекеры будто повторяли старые добрые трекеры позднего DOS и Windows 98. Linux-разработчики стали клонировать *Fast Tracker* и *Impulse Tracker*. Особо удачными стали два: *Schism Tracker* (http://nimh.org/schism) и *Cheese Tracker* (www.reduz.com.ar/cheesetronic/index.php) – причем первый выглядит почти точной копией оригинального *Impulse Tracker* и может работать в окне или в полноэкранном режиме, а интерфейс *Cheese Tracker* основан на *Qt 3*, и этот трекер работает только в окне (**рис. 1**).

В этой статье я расскажу о работе с трекерами на примере *Schism Tracker*. Почему не *Cheese*? По моим наблюдениям, *Schism Tracker* более правильно воспроизводит модули и, как сказано выше, практически повторяет собой *Impulse Tracker*, а значит, пользователь может



› **Рис. 1. Окно *Cheese Tracker*.**

использовать многочисленные статьи о последнем, в том числе и руководство к *Impulse Tracker* на русском: http://trackers.pp.ru/info/track.php?list=it2ug. Другие ресурсы перечислены во врезке «Полезные ссылки».

Наконец, ответим на последний вопрос – в каких программах воспроизводятся трекерные модули? Между прочим, самой популярной из них была игра *Unreal* (первая часть) – там использовалась музыка в формате *Impulse Tracker*. А вообще, модули умеет играть любой плейер, к которому есть подключаемый модуль на основе библиотеки *Modplug*. Это *XMMS*, *Audacious*, *BMP*, *Winamp*. Эти же плейеры могут конвертировать модули в WAV, используя расширение *Disk-writer*.

А теперь начнем работу с *Schism Tracker*.

Интерфейс Schism Tracker

Интерфейс *Schism Tracker* почти полностью повторяет интерфейс *Impulse Tracker*. Тот, в свою очередь, был основан на интерфейсе другого трекера – *Scream Tracker 3*. Во времена DOS такой нестандартный интерфейс не был чем-то особенным. Напротив, интерфейс *Impulse/Schism Tracker* довольно удобен, если к нему привыкнуть. Управление трекером осуществляется в основном с клавиатуры, хотя поддерживается и мышь. В DOS *Impulse Tracker* работал в текстовом режиме экрана, хотя впечатление складывается совершенно обратное. В самом деле, все элементы управления – поля ввода, кнопки, ползунки и прочее были выполнены в текстовом режиме, разве что шрифт использовался особый. В *Schism Tracker* тоже применяется подгружаемый шрифт (и даже имеется встроенный редактор шрифтов), но вместо текстового режима трекер работает в графическом окне. Как уже говорилось, можно переключаться и в полноэкранный режим.

Интерфейс *Schism Tracker* состоит из экранов, между которыми можно переключаться либо клавишами, либо с помощью главного меню, которое вызывается нажатием на **Esc**. Изучать интерфейс лучше в ходе работы, поэтому сейчас я пошагово опишу, как создать музыкальную композицию в *Schism Tracker*. Безусловно, ноты за вас я придумывать не стану.

Создание нового модуля

Чтобы создать новый модуль – файл с музыкальной композицией – нажмите Ctrl-N. Появится диалоговое окно, в котором спрашивается, что делать с паттернами, сэмплами, инструментами и порядком воспроизведения паттернов из текущей песни. Можно оставить их (keep) как шаблон для нового модуля, а можно начать модуль с чистыми параметрами (clear). По умолчанию выбрано последнее.

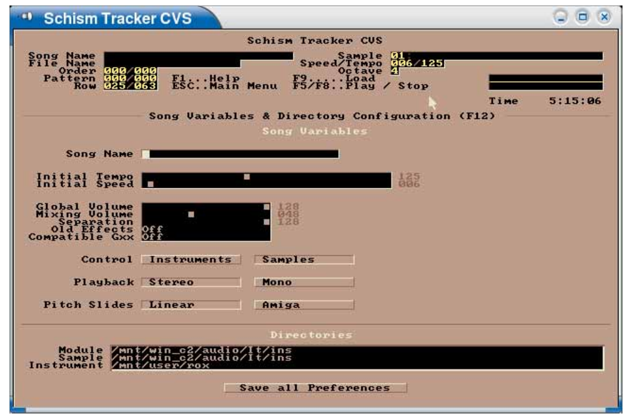
После этого в нашем распоряжении будет новый модуль и мы сможем задать различные характеристики. Нажмите F12, чтобы попасть в экран настроек песни:

Полезные ссылки

» http://trackers.pp.ru/info/track.php?list=it2ug
Руководство к Impulse Tracker на русском языке

» http://trackers.pp.ru/links/
Информация о том, откуда брать сэмплы, готовые модули и «сопутствующие товары»

» http://www.modarchive.com/
Более 34 000 трекерных готовых модулей. Помимо прочего, из них можно брать и сэмплы (указывая авторство).



Здесь можно управлять такими свойствами песни, как ее название (поле Song name), начальный темп (Initial temp), громкостью (общей и микширования). Кроме того, здесь же указываются каталоги, где расположены сэмплы, инструменты и модули.

Значение темпа задается в BPM – beats ber minute, то есть количество ударов в минуту. Чем выше это значение, тем быстрее воспроизводится песня. Для справки – обычный ритм техно или хауса – 180 ударов в минуту.

Чтобы снабдить песню какой-либо текстовой заметкой, нажмите Shift-F9 и введите или отредактируйте текст (поддерживается только латиница). Некоторые плейеры умеют его отображать, некоторые – нет.

Сохранение и загрузка

Чтобы сохранить модуль, надо нажать Ctrl-S. Если модуль не был ранее сохранен, то появится экран «Сохранить как» – точно такой же можно вызвать в любое время, нажав F10. Имя сохраняемого файла указывается в поле Filename. Введя имя, нажмите Enter, и файл будет сохранен.

Справа от списка каталогов вы видите кнопки, с помощью которых можно выбрать формат сохраняемого файла. Auto сохраняет модуль в том формате, в котором он был открыт. *Schism Tracker* при сохранении поддерживает модули нескольких форматов – **IT214** (*Impulse Tracker 2.14*), **XM** (формат *Fast Tracker*), **S3M** (*Scream Tracker*) и **MOD**. Если вы сохраняете только что созданный файл, то выбор Auto приведет к сохранению в основном формате *Schism Tracker* – **IT214**.

Также среди форматов есть **WAV** – выбрав эту кнопку, вы сможете «сконвертировать» ваш модуль в **WAV**, чтобы потом обработать его в какой-нибудь другой программе или записать на CD.

Загрузка файлов происходит по нажатию на F9. *Schism Tracker* уме- »

Кто подставил *Impulse Tracker*?

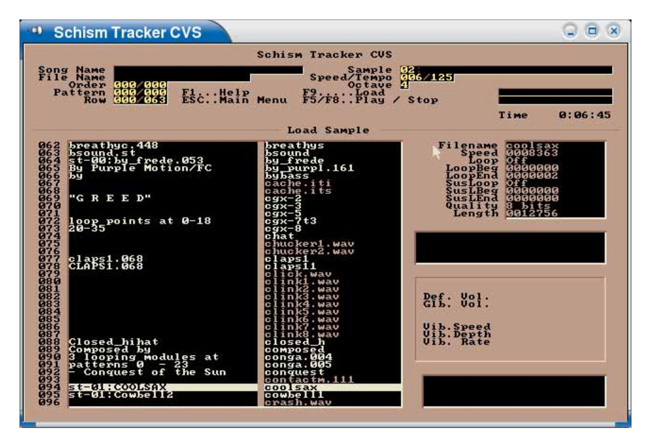
Примечательно, что полноценный вывод композиции в WAV стал причиной прекращения разработки *Impulse Tracker*. *Impulse Tracker* распространялся как Freeware. Вместе с *Impulse Tracker* поставлялся драйвер для вывода в WAV, однако с ограниченными возможностями – он записывал только моно-файлы. Разработчик трекера, австралиец Джеффри Лим [Jeffrey Lim], отдельно продавал полнофункциональную версию этого драйвера. После того, как пираты сделали коммерческую версию доступной для всех желающих, Лим объявил о прекращении разработки своего трекера. Исходные тексты написанного на ассемблере *Impulse Tracker* были закрыты, так что о продолжении разработки не могло быть и речи.

» ет загружать не только файлы своего «родного» формата – то бишь формата *Impulse Tracker*, но и модули от других трекеров, например, того же *Fast Tracker*.

Загрузка сэмплов в песню

Теперь, когда вы знаете, как создавать новый модуль и сохранять его, давайте посмотрим, как загружать сэмплы, которыми вы будете играть мелодию. Я не буду рассказывать о трекерных инструментах – это отдельная тема и подробно о ней вы можете прочтесть в руководстве к *Impulse Tracker*. Для начала хватит и обычных сэмплов.

Сэмплом может быть, во-первых, любой **wav**-файл с разрядностью 8 или 16 бит. Замечу, что продаются (пиратские) диски с сэмплами, где файлы имеют расширения **.wav**, хотя на самом деле это **MP3**. Так вот, MP3-сэмплы *Schism Tracker* не понимает. Зато понимает сэмплы многих других форматов (некоторые трекеры имеют свои собственные форматы сэмплов). Кроме того, *Schism Tracker* умеет «заходить» в трекерные модули, как в каталоги, и позволяет вам загружать чужие (или свои) сэмплы прямо из модулей.

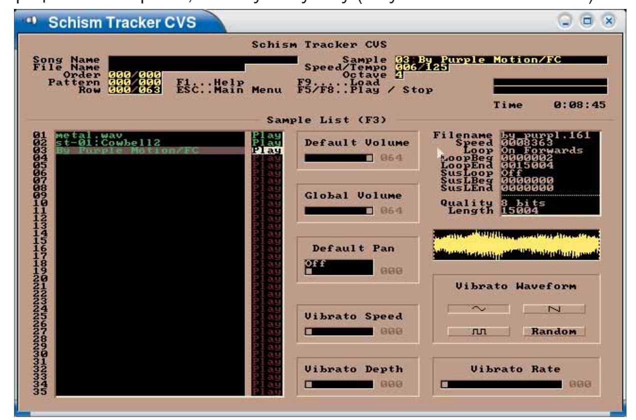


Нажмите F3. Если ни один сэмпл не был загружен, вы попадете прямо в экран с выбором файлов. Если же какие-то сэмплы уже загружены, то нажатие на F3 вызывает экран со списком сэмплов. Установка курсора на сэмпл делает его текущим – именно этот сэмпл будет выбран для записи им нот партии, которую вы редактируете в экране паттерна (об этом чуть позже).

Нажатие Enter на сэмпле в списке снова переносит вас в экран выбора файлов – так можно заменить уже загруженный сэмпл на другой. А чтобы загрузить новый сэмпл, следует нажать Enter на пустой строке в списке.

И в списке файлов, и в списке загруженных сэмплов вы можете опробовать, как звучит сэмпл. Для этого просто играйте на буквенной части клавиатуры.

В списке загруженных сэмплов можно настраивать их параметры – громкость, панораму (расположение в стерео-пространстве), тип вибрации и ее скорость, частоту и глубину (по умолчанию выключено).

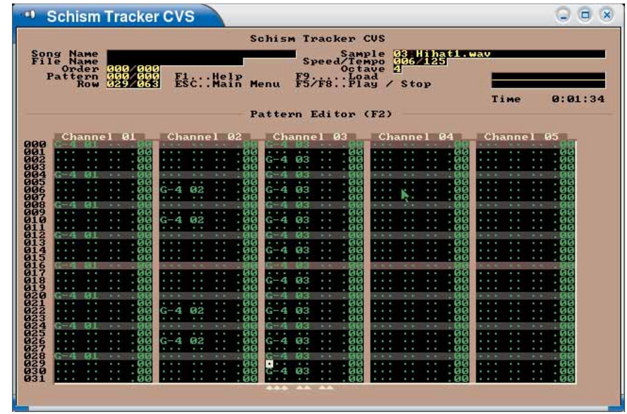


Кроме того, часть сэмпла можно зациклить, превратив в петлю, что полезно для всяких синтезаторных «подушек», которые должны длиться произвольно долго, а не заканчиваться, как только сэмпл подходит к концу. *Schism Tracker* позволяет также производить над сэмплами некоторые операции – реверс, увеличение громкости и так далее.

Загрузите сэмпл и выберите его в списке, установив на нем курсор. Теперь перейдем к самому интересному – напишем этим сэмплом партию, мелодию.

Редактор паттернов

Нажмите кнопку F2, чтобы попасть в экран редактирования паттерна:



Вы видите, что он разделен на дорожки – каналы. В свою очередь, каждый канал состоит из строк – рядов. А в ряде – четыре колонки. Всё, что можно туда помещать, вводится с клавиатуры. Первая колонка содержит в себе ноту и октаву. Октавы переключаются клавишами * и /, а ноты вводятся нажатием буквенных клавиш. Вторая колонка – это номер сэмпла.

Как вы могли заметить, в экране сэмплов (F3) сэмплы пронумерованы. Допустим, вы прописали партию сэмплом номер **2**, а потом решили, что сэмплом номер **5** эта партия будет звучать лучше. Что делать – переписывать всю партию другим сэмплом? Нет, достаточно изменить номер сэмпла в соответствующей колонке. Это можно сделать как отдельно в каждом ряду (вручную), так и для нескольких рядов сразу, следующим образом:

- Выделите ряды (Shift-стрелки, либо Alt-B – начало выделения, Alt-E – конец выделения).
- В экране сэмплов сделайте текущим сэмпл, на который вы хотите сменить сэмпл выделенных рядов.
- В экране паттернов нажмите Alt-S.

Но вернемся к разбору колонок ряда. Третья колонка – настройки панорамы и громкости. Что именно мы настраиваем, переключается клавишей ~ (тильда). На деле задавать громкость и панораму в этой колонке надо лишь тогда, когда вы хотите регулировать эти параметры динамически, по ходу воспроизведения нот. Например, чтобы создать эффект «бегающего» из уха в ухо звука, надо изменять значения панорамы. 32 – центр. Чем меньше тридцати двух, тем более сэмпл звучит слева, а чем больше 32, тем правее. Меняя эти значения для каждой ноты, мы получаем эффект «из уха в ухо», хотя для того же можно использовать и специализированный эффект, прописываемый в колонке эффектов. Но эффекты – тема обширная, выходящая за рамки этой статьи. Подробнее о них читайте в руководстве к *Impulse Tracker*.

Если же вы хотите задать статичные настройки для громкости и панорамы, то существует микшер, вызываемый по F11 (там каждому каналу можно выставить панораму и громкость). Повторное нажатие F11 в этом экране переключает микшер с настройки панорамы на настройку громкости. Также можно настраивать громкость и панораму отдельно для каждого сэмпла в экране по F3.

Последняя колонка – колонка эффектов. Ее мы пропускаем, нам надо успеть еще много в чем разобраться. Во-первых, как добавлять новые паттерны и как переключаться между ними? Для этого служат клавиши плюс и минус на цифровой части клавиатуры. Плюс переносит вас на паттерн вперед, минус – на паттерн назад. Нажатие плюс создает новый паттерн после текущего, если текущий паттерн – пос-

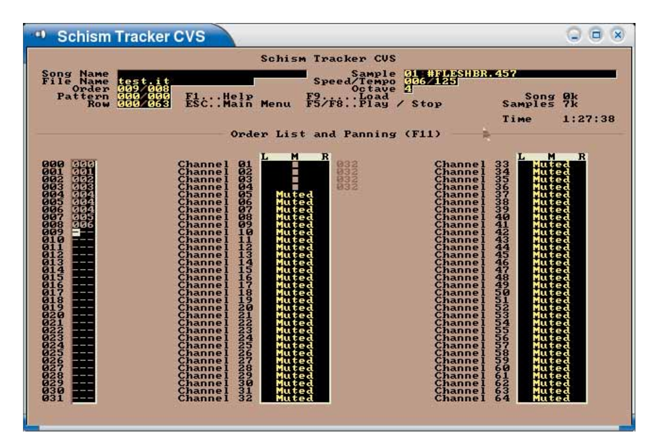
ледний. Удалять сами паттерны нельзя, можно удалять только их содержимое. Настроить параметры текущего паттерна можно в окне, вызываемом клавишей F2, нажатой в экране паттерна. Доступны такие параметры, как длина паттерна (количество рядов), опции подсветки, основная октава, шаг курсора и так далее.

Записав партию на одном канале, можете писать другую партию на другом канале. Доступно 64 канала – этого более чем достаточно. Ноты играют во время набора вами мелодии. Чтобы прослушать паттерн с места курсора, нажмите F7. Чтобы прослушать весь паттерн, нажмите F6. Паттерн играется в цикле, пока не прервете его клавишей F8.

Завершая рассказ об экране паттерна, приведу некоторые полезные сочетания клавиш. Alt-C, Alt-P – копировать/вставить выделенные на каналах данные. Alt-Q/A – поднять или опустить выделенные ноты на полтона.

Порядок воспроизведения

Клавиша F11 переносит нас в экран Order List (совмещенный с микшером громкости и панорамы):



Теперь нас интересует список слева. Это и есть список, задающий порядок воспроизведения паттернов. Сюда вводятся номера паттернов в той последовательности, в какой они должны воспроизводиться. Можно добавлять номера, вставлять в произвольное место, удалять, дублировать. Таким образом, вы собираете песню из отдельных паттернов. Чтобы запустить всю песню на воспроизведение, нажмите F5. Чтобы остановить воспроизведение, нажмите F8.

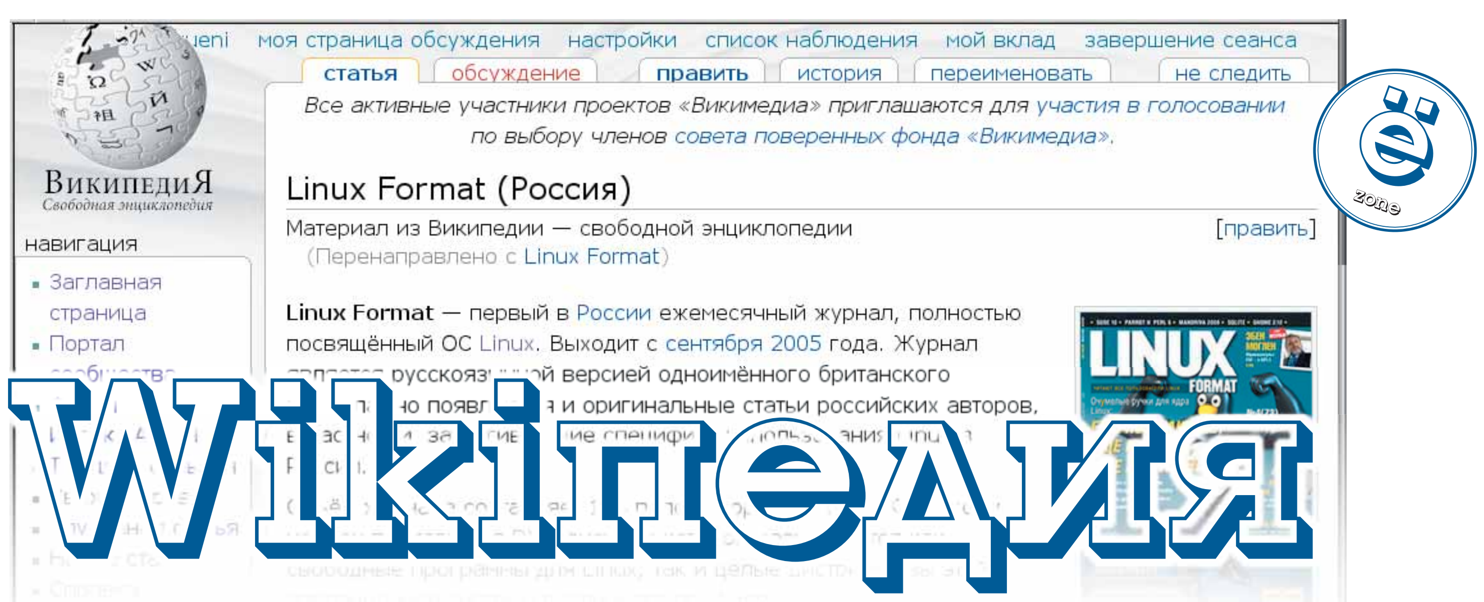
Заключение

Вот, в принципе, и все основы трекерной грамотности. Подчеркиваю: основы! Чтобы использовать трекер на полную катушку, надо прочитать к нему руководство. И – практиковаться.

Какая музыка получается в трекере лучше всего? Да любая, хотя чаще можно встретить jungle, drum'n'bass, различные виды techno и trance. Качество звучания модулей напрямую зависит от качества использованных сэмплов. Восьмибитные и с низкой частотой оцифровки звучат грязно, шипят.

Напоследок поговорим о «союзе» трекеров и партий, записанных вживую (гитара, вокал и так далее). В принципе, это возможно (хотя такие партии придется записывать в какой-нибудь другой программе), но трекер – это прежде всего средство для создания музыки на компьютере, а не микширования «живых» партий. Принято делать модули сравнительно небольшими, а кто будет качать его, если он содержит в себе живые партии и занимает сто мегабайт? Обычный размер модуля – не более двух мегабайт, а чаще всего до мегабайта, пара сотен килобайт. LXF

Через месяц Мы поговорим о барабанных машинах и виртуальных синтезаторах.



Wikiпедия

Получив секретное задание от редакции журнала, **Евгений Балдин** внедрился в сообщество Википедии – и сам не заметил, как стал завзятым Википедистом.

Всё есть статья!

Интересно читать энциклопедию, но гораздо интереснее эту энциклопедию создавать. Делиться знаниями – это ни с чем не сравнимое удовольствие и доступно оно только избранным. Как, вы ещё не Википедист? Подумайте над этим на досуге.

Получив добро от редакции *Linux Format*, я постарался внедриться в сообщество, формирующиеся вокруг российского отделения Википедии (**http://ru. wikipedia. org**). За время внедрения было сделано около двух сотен правок и написана одна «хорошая статья». Попытка понять логику Википедии также реализовалась в виде статьи. Что, впрочем, не удивительно, так как в основе Википедии лежит статья – это сама сущность проекта.

Чем не является Википедия

Википедия – это не энциклопедия. На информацию в Википедии нельзя ссылаться как на источник надёжных данных. Их точность нельзя гарантировать. Возможно, в будущем будет создан авторитетный инструмент рецензирования и проверки, но пока его нет. С другой стороны, ссылки на статьи в Википедия вполне допустимы для вводного ознакомления с вопросом.

Википедия – это не личный дневник. Знания в Википедии обезличены. Это не форум и не новостная газета. Википедия – не рекламная площадка. Нейтральность точки зрения здесь является единственно возможной линией поведения.

Википедия – это не файлообменник и не домашняя страничка. В любую минуту любые данные могут быть отредактированы. Ничто не гарантирует незыблемости.

В Википедии нет цензуры, поэтому содержание некоторых статей может сильно противоречить чьей-либо точки зрения. Так как Википедия – не поле боя, то сообществом Википедии была выработана процедура разрешения конфликтов: «Википедия: Разрешение конфликтов».

Подробнее про упомянутое выше и про многое другое можно прочитать в статье «ВП:ЧНЯВ», то есть «Википедия:Чем не является Википедия».

Чем полезна Википедия

Чем Википедия полезна для читателей? Многоязычностью, гипертекстом и возможностью в любой момент стать писателем. Как правило,



› **Страничка, посвящённая *Linux Format*.**

любая статья имеет аналоги на других языках и на них можно сослаться, используя механизм интернациональных ссылок («Википедия: Интервики»). Гипертекст позволяет бродить до полного прояснения ситуации. Входной порог для внесения простых правок в текст очень низок, если вообще его можно измерить.

Тексты в Википедии предоставлены для свободного использования и правки («Википедия:Правовая основа»). Статьи Википедии публикуются под лицензией GNU FDL (**http://www. gnu. org/licenses/fdl. html**) со всеми вытекающими последствиями. В частности, все производные от этих текстов должны иметь ту же лицензию, что и сами тесты.

Информация, занесённая в Википедию, имеет тенденцию к постепенному развитию. Специализированную информацию можно хранить на Википедии без особых опасений. Спорных статей, вызывающих конфликты, не так уж и много, а нейтральные статьи достаточно устойчивы.

В помощь новичкам

Перед редактированием статьи полезно зарегистрироваться. Для этого достаточно щелкнуть ссылку в правом верхнем углу «Представиться системе» (статья «Википедия:Регистрация»). В этом случае можно

получить доступ к настройкам, к журналу своих изменений и к возможности наблюдать за интересующими вас статьями.

В поле навигации слева есть «волшебная» ссылка под названием справка («Википедия:Справка»). Там есть всё. Хотя это не так интересно, как искать ответы методом «тыка», но лучше заглянуть туда до редактирования статьи, хотя бы для ознакомления. Справочный материал объединён в одну категорию «Категория:Википедия:Справка».

Если не терпится что-то набрать, то полезно просмотреть рекомендации по оформлению статей с учётом русскоязычных традиций – «Википедия:Оформление статей».

Что делать?

Вопрос вечный, но в данном случае на него легко ответить:

›› Можно дорабатывать уже существующие статьи. Это редкий случай, когда полировка текста не сдерживается никакими временными рамками. Текст может правиться столько, сколько нужно и даже сверх того.

›› Можно добавлять рисунки к уже существующим статьям. Часто один рисунок стоит сотни слов.

›› Можно перевести уже существующие статьи с других языков. По числу статей англоязычная *Wikipedia* превосходит русскую Википедию на порядок.

›› Ну и, наконец, можно создавать новые статьи. Предварительно с помощью поиска необходимо убедиться, что аналогичная статья ещё не создана. Имеет смысл заглянуть и в иноязычные Википедии.

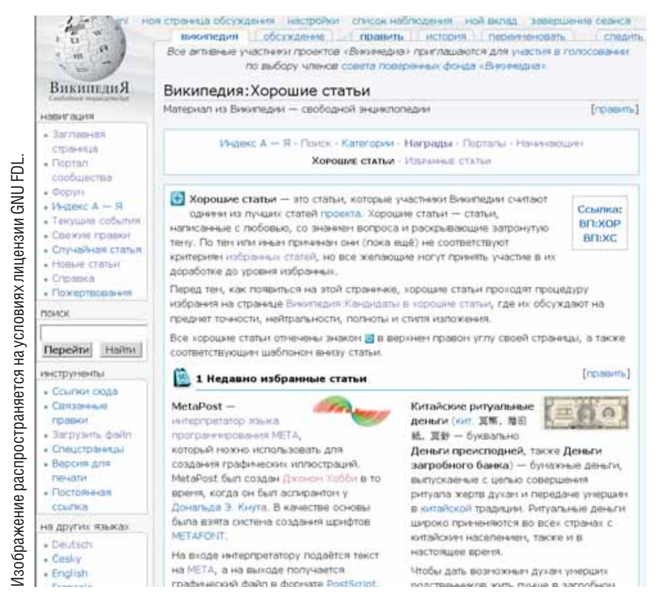
Правила хорошего тона

Правила, следование которым, предположительно, есть хороший тон:

›› Следует уважать авторские права. Бездумное копирование текста, защищённого значком ©, или любого авторского текста без разрешения автора – моветон. Информация в Википедии должна быть свободна для распространения и использования.

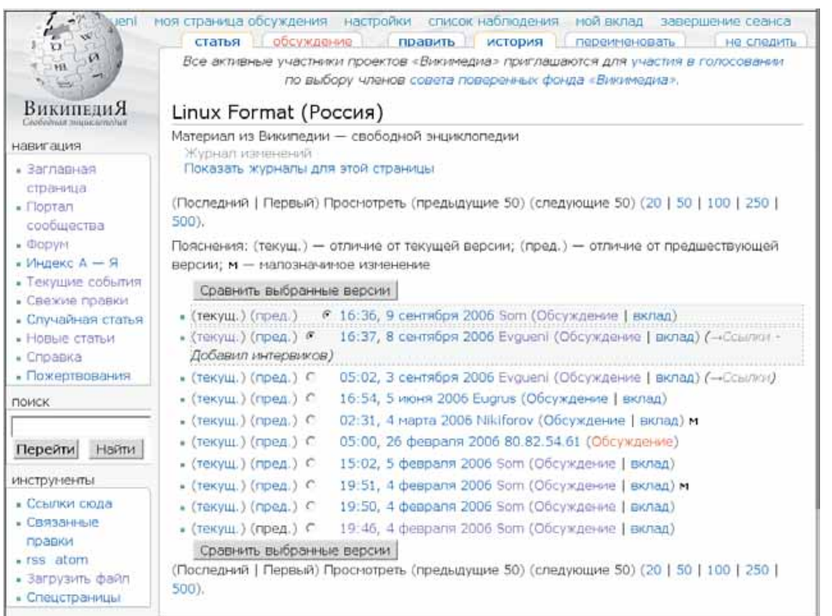
›› Статьи должны быть информационными, а не эмоциональными. Не следует выдавать своё виденье за общепринятое. ОРИС[1] не место в Википедии. НТЗ (нейтральная точка зрения) является одним из столпов Википедии («Википедия:Пять столпов»).

›› Технология Википедии поощряет правку статей. Делайте это смело (статья «Википедия:Правьте смело») без боязни что-либо испортить. Всегда можно произвести «откат» (статья «откат») к предыдущей версии. Естественно, действовать следует прилично. Хулиганские выходки в логику Википедии не укладываются.



› «Хорошие статьи». Лучшие статьи Википедии выбираются путём выдвижения и голосования.

[1] Акроним «ОРигинальное ИСследование», подробнее в статье «Википедия:Об оригинальных исследованиях».



› История изменений статьи *Linux Format*.

›› К сделанным правкам следует добавлять комментарии[2] в специальном поле формы редактирования «Краткое описание:». Если правки мелкие (исправление орфографии или «ёфикация» [3]), то это тоже следует отметить, проставив галочку в пункте «Малое изменение». Позже по журналу изменений будет проще понять логику правки.

›› Пишите о том, что вы знаете и о том, в чём заинтересованы. Не следует писать тексты с ненавистью – это заметно.

›› Все эмоции выносятся на страницу обсуждения, которая есть у каждой статьи. Подпись при высказывании своего личного мнения на странице обсуждения (четыре знака тильды «~~~~») является обязательным элементом.

›› Википедия – гипертекстовое образование, поэтому везде, где можно, следует вставлять ссылки на другие статьи. Список источников в конце статьи увеличивает её ценность, так как позволяет продолжить интересующее читателя исследование.

›› Просмотрите статью «Википедия:Правила и указания» перед тем, как начать исправлять что-то серьёзное.

Логика Википедии

Логика Википедии – *всё есть статья*. У каждой статьи есть страница обсуждений и журнал изменений. «Все ходы записаны» – любые исправления хранят имена своих авторов.

Вики-разметка

Вики-разметку нельзя назвать изощрённой. Она очень проста – это её и сильная, и слабая стороны. Простые тексты создаются без усилий, а сложное форматирование, как следствие, недостижимо. Базовые правила перечислены в статье «Википедия:Как править статьи».

Структура

Структура текста формируется с помощью знака равно «=»:



```
== Раздел ==
=== Подраздел ===
==== Подподраздел ====
```

При вставке подобной конструкции появляется возможность редактировать только конкретный раздел/подраздел, а не весь текст. Длинные тексты необходимо структурировать не только для удобства редактирования, но и для удобства навигации, так как при отображении статьи автоматически создаётся оглавление.

»

[2] Стандартный совет пользователю любой системы контроля версий.

[3] Замена буквы «е» на букву «ё» везде, где это необходимо.

Википедия:Вавилон

ru — Для этого участника русский язык является родным.

en-2 — This user is able to contribute with an intermediate level of English.

Ашыпка — Разрешаю всем поправлять мои орфографические и грамматические ошыбки.

Ё — Этот участник выступает за обязательное использование буквы ё в русском языке


Этот участник является сторонником свободного ПО


Этот участник использует Linux


Этот участник использует Debian GNU/Linux


Этот участник использует PalmOS

Изображение распространяется на условиях лицензии GNU FDL.

› Выбранные мной «юзербоксы».

Набор текста

Обычный текст должен начинаться без отступа, абзацы разделяются пустой строкой. Если первый символ в строке – пробел « », то текст считается предварительно отформатированным и отображается шрифтом фиксированной ширины без автоматического переноса строк. Этот приём следует использовать для представления исходного кода программы. Этого же эффекта можно достигнуть, заключив текст между тэгами <pre> «текст» </pre>. Для экранирования википодобных конструкций в тексте можно использовать тэги <nowiki> </nowiki>.

Списки и отступы

Место в начале строки вообще является особым. Если первым символом оказывается двоеточие «:», то при просмотре абзац форматируется с отступом – удобно при ответе на чей-то вопрос. Пункты нумерованных списков отмечаются знаком решётки «#» в начале строки. Для ненумерованных списков используется знак звёздочки «*». Эти знаки можно комбинировать, создавая конструкции из сложных вложенных списков и отступов.

Ссылки

Ссылки между статьями Википедии создаются с помощью открывающих и закрывающих двойных квадратных скобок:

```
[[имя статьи, на которую идёт ссылка |
отображаемый текст]]
```

Вертикальная черта «|» разделяет ссылку и отображаемый текст. Если статьи, на которую ведет ссылка, не существует, то при отображении ссылка выделяется красным цветом, как бы приглашая дописать эту статью. Внешние ссылки ограничиваются одинарными квадратными скобками и не требуют разделителя между ссылкой и отображаемым текстом:

```
[http://linuxforum. ru/?showforum=57 Форум
Linux Format]
```

Названия статей

«Как правильно назвать статью?» – не такой уж и тривиальный вопрос. Во-первых, необходимо следовать правилам русского языка («Википедия:Именование статей» и «Википедия:Имена»), а во-вторых, учитывать ограничение движка Вики («Википедия:Соглашение об именах (технические ограничения)»). Неправильно называть статьи – моветон[4].

Для коррекции названия статьи можно применить шаблон

```
{{title|правильное название}}
```

Таблицы

О том, как делать несложные таблицы, рассказано в статье «Википедия: Как делать таблицы». Это не совсем тривиальная операция, но всё же лучше, чем ничего.

Формулы

Для набора формул полезно присмотреться к статье «Википедия: Формулы». Если Вы знакомы с нотацией *TeX*, то набор формул будет естественен (см. LXF85). Всё, что заключается между тэгами $$, трактуется как *TeX*-нотация. Если слово *TeX* вам не знакомо, а формулы набирать хочется, то будет лучше разобраться с этим понятием, так как до сих пор нет лучшего способа текстового представления формул.

Изображения

Ничто так не украшает статью, как пара-тройка картинок по теме. Лучше один раз увидеть, чем…

Информация о технологии размещения изображений подробно изложена в статье «Википедия:Изображения». Загрузка файла происходит из статьи «Служебная:Upload». Ссылка на неё расположена слева на служебной панели в разделе «Инструменты».

Предпочтительными являются растровые форматы PNG (статья «png») и JPEG (статья «jpeg»). Загруженная картинка – это тоже статья, обязательным элементом которой является лицензия. В случае отсутствия информации о лицензии изображение удаляется из Википедии. Наиболее популярным типом лицензии является лицензии от Creative Commons (статья, естественно, «Creative Commons» или LXF81[5]). Простейшим источником лицензионно чистых изображений могут быть:

- Ваша цифровая камера – никаких проблем с копирайтами.
- Ваша любимая программа рисования и ваш талант художника.

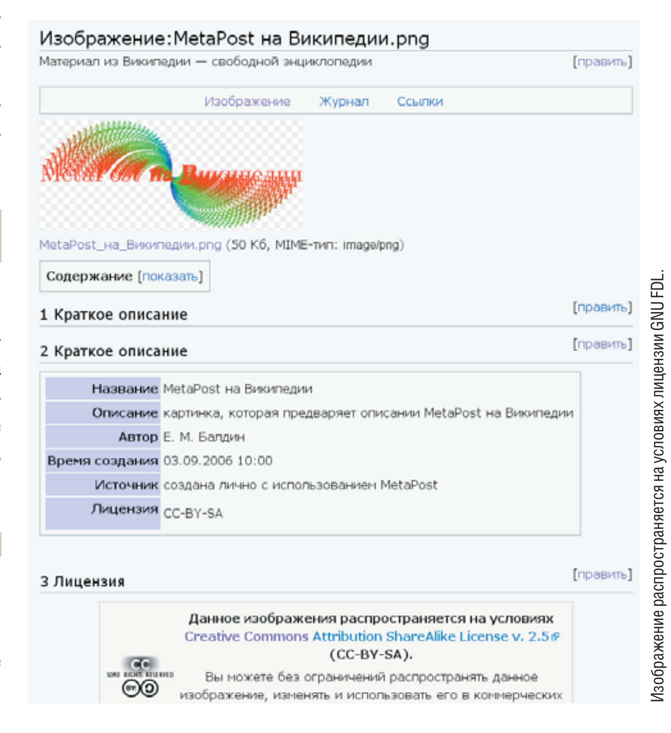
Брать изображения откуда-то ещё можно только с разрешения правообладателя – такова логика Википедии.

Население Википедии

В Википедии есть несколько групп антропоморфных существ: анонимусы, зарегистрированные пользователи, администраторы («Википедия: Администраторы») и бюрократы («Википедия:Бюрократы»).

Все участники равны, но некоторые «равнее других» – иерархия всё же наличествует. Перестать быть анонимусом может каждый – путём регистрации. А вот чтобы стать администратором – необходима помощь бюрократа.

Также есть отдельная категория псевдопользователей – боты



Изображение распространяется на условиях лицензии GNU FDL.

› Представление картинки на Википедии. Теперь, когда её закачали и указали лицензию, на неё можно ссылаться..

[4] Но иногда, если требуется по смыслу, это правило можно обходить. Например, созданная мной на Викитеке категория «НЕтрадиционная НАУКА» лучше отражает смысл статей, принадлежащих этой категории, чем в случае традиционного именования.

[5] Подробно лицензии Creative Commons были рассмотрены в статье Йона Филипса «Лицензия на творчество» LXF71(81) июль 2006. Статья под лицензией CC-BY-NC доступна так же и в электронном виде по ссылке http://www.linuxformat.ru/mag/creativecommons.pdf

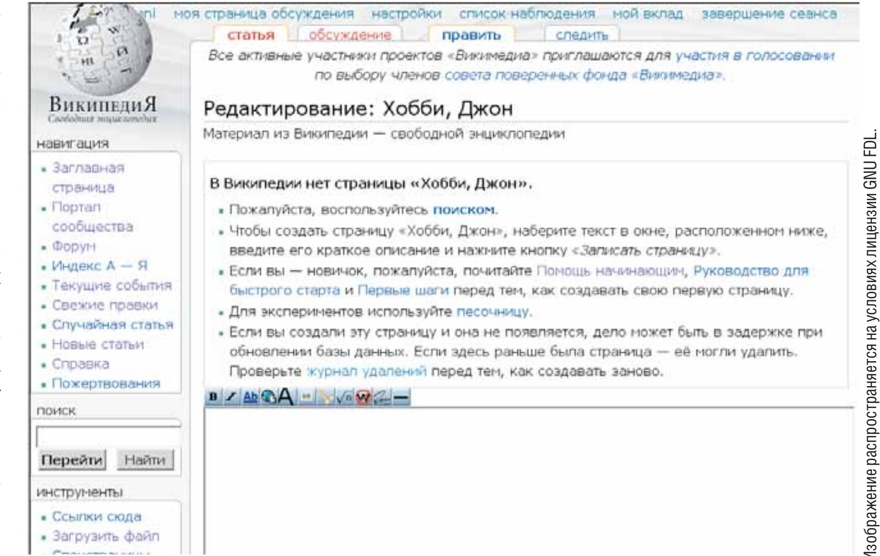
(«Википедия:Бот»). Это автоматические программы, которые выполняют рутинные операции по исправлению технических ошибок в статьях.

Администраторы имеют чуть больше прав, чем простые пользователи. Они имеют право удалить статью («Википедия:Удаление страниц») или, наоборот, запретить её правку («Википедия:Частичная защита страниц»).

Форумы, обсуждения и проекты

Википедия не предназначена для общения. Да и сообщество не очень-то и большое: активных участников во всём русском сегменте около двух сотен. Но совсем без взаимодействия не обойтись. Для продвижения идей по улучшению качества Википедии в жизнь создаются страницы, предназначенные для обсуждения. Примером интересного проекта является «Википедия:Кандидаты в хорошие статьи», где происходит обсуждение статей для присвоения им высокого звания «Хорошая статья» («Википедия:Хорошие статьи»). Все подобного рода страницы имеют приставку «Википедия:» в начале своего имени, то есть относятся к пространству имён «Википедия».

Общение по делу также может вестись на странице обсуждения статьи или на личной странице обсуждения участника.



› **Пример того, что происходит, когда пытаешься попасть на страницу, которая ещё не написана. Её просто предлагают дописать.**

Порталы и категории

Кроме обычных статей и статей обсуждения, есть ещё служебные страницы. Цель служебных страниц – облегчение поиска или развитие Википедии.

Одним из типов служебных страниц являются Порталы («Википедия: Порталы»). Порталы служат основными страницами для введения в предметную отрасль, например, в физику, фантастику, аниме или шахматы. Страницы Порталов относятся к пространству имён «Портал».

Для категоризации Википедии используются страницы-категории («Википедия:Категории»). Такие страницы всегда начинаются с приставки «Категория:». Для включения статьи в определённую категорию необходимо внутри неё добавить ссылки вида:

```
[[Категория:TeX]]
[[Категория:Прикладное программное обеспечение]]
[[Категория:Свободное программное обеспечение]]
```

Это список категорий, в которые входит статья «MetaPost». После вставки такого текста в конце статьи при просмотре появляются ссылки на указанные категории. На странице категории, кроме обычного текста, расположен список всех страниц, которые на неё ссылаются. Категории являются обычными страницами, поэтому могут ссылаться на другие категории – получается естественная иерархия. Корнем всех остальных Категорий является «Категория:Всё».

Шаблоны

Шаблоны – это класс особых страниц, которые можно вставлять в другие страницы («Википедия:Механизм шаблонов»). Это очень мощный механизм, позволяющий унифицировать многие рутинные действия. Для вставки имя шаблона необходимо заключить в двойные фигурные скобки: {{Имя шаблона}}

Шаблоны можно использовать как пометки: например, если случайно была создана статья с неправильным заголовком, то для её удаления надо просто добавить шаблон запроса на быстрое удаление: {{delete|причина удаления}}. Эта метка для администратора Википедии, который уже решает, действительно ли надо удалять плохую статью или нет. О механизме удаления подробно написано в статье «Википедия:Удаление страниц».

Шаблоны можно использовать для улучшенного форматирования текста, например, так можно вставить цитату: {{Начало цитаты}} тело цитаты {{Конец цитаты}}.

Довольно интересно применение шаблонов в качестве «юзербоксов» («Википедия:Юзербоксы»). «Юзербоксы» встречаются на личных страничках участников и характеризуют пристрастия автора. Например, шаблон

```
{{Участник за букву ё}}
```

сразу даёт понять, что автор при написании своих статей использует упомянутую букву там, где она должны быть по праву.



› **Моя статья на Википедии. Это «хорошая статья».**



› **Английская версия статьи *MetaPost* – работа Интервиков.**

2 MetaPost [править]

Значительно доработал статью по MetaPost по мотивам моей публикации в Linux Format. Специально для статьи сделал и закачал несколько картинок-примеров с кодом в описании. Буду премного благодарен за советы как её улучшить. Викитехнические подсказки тоже будут очень кстати. — Evgueni 11:54, 3 сентября 2006 (UTC)

Гхмм, хочу комментариев или эта спецтема совсем не интересна? — Evgueni 15:16, 4 сентября 2006 (UTC)

- Жестковато -- лучше большую часть перенести в учебник, а здесь оставить пару характерных "рекламных" примеров с исходниками. Maxim Razin 16:35, 5 сентября 2006 (UTC)

Это только введение с минимальными примерами. Учебник гораздо больше будет. — Evgueni 04:59, 6 сентября 2006 (UTC)

- Не знаю, я ⊕ **За**. Мне статья понравилась. Как мне кажется, в викиучебник переносить не надо, здесь не учебник, а скорее описание формата и несколько примеров. Кстати, я примеры сделал в галерею, мне кажется, так компактнее и лучше :) Калий 17:11, 5 сентября 2006 (UTC)

Я думал на счёт тега галерея, но мне не понравилось форматирование картинок по умолчанию. С другой стороны, возможно, лучшего от такого простого формата ожидать и не приходится. Придётся добавить ещё два примера :) — Evgueni 04:56, 6 сентября 2006 (UTC)

- Да, кстати :). В линуксформате понравились Ваши статьи (в учебнике которые). Не желаете ли перенести их в Викиучебник? Калий 17:41, 5 сентября 2006 (UTC)

Права на текст в Linux Format возвращаются к автору через пол года после публикации, так что это возможно будет сделать где-то в районе января, если время будет. Собственно говоря, это статья по мотивам **Введения** из цикла — здесь пол года уже прошло. Существует проблема, что Wiki меня по многим параметрам не устраивает. На Форуме Викиучебника я пытаюсь понять что мне будет удобно. Пока решения кроме публикации, как pdf+архив исходников не вижу. — Evgueni 04:56, 6 сентября 2006 (UTC)

- ⊕ **За**, хорошая статья. stassats 16:07, 8 сентября 2006 (UTC)
- ⊕ **За**. Никогда раньше не мог подумать, что так просто можно генерировать изображения. Обычно я писал для этого отдельные программы. kmeaw 17:35, 8 сентября 2006 (UTC)

› Обсуждение на присуждении моей статье *MetaPost* высокого звания «хорошая статья».

Все наиболее часто встречающиеся шаблоны перечислены в статье «Википедия:Шаблоны».

Интервики

Интервики – это механизм создания ссылок между различными Вики-проектами. В Википедии существует механизм связывания статей на различных языках друг с другом. В более-менее развитых статьях на панели слева внизу в разделе «На других языках» перечисляются языки Википедии, в которых есть статьи на эту же тему – это работа интервиков. Интервики вставляются в тело статьи и имеют примерно следующий вид:

```
[[en:MetaPost]]
[[de:MetaPost]]
[[ja:MetaPost]]
<!-- и так далее -->
```

Первые две буквы соответствуют коду языка по стандарту ISO 639, а затем идёт название статьи на языке оригинала. В данном случае на всех языках для названия статьи используется одно и то же слово. Нет необходимости рыскать по всем Википедиям для поиска соответствующих статей. Достаточно вставить один интервик, например, на англоязычную статью, а остальную работу по дополнению ссылок доделают «боты».

Кроме создания межъязыковых связей, интервики позволяют ссылаться на «братские» Вики-проекты. Подробности описаны в статье «Википедия:Интервики».

Механизм межъязыковых ссылок превращает Википедию из простой локальной «недоэнциклопедии» в хранилище знаний с окнами во всё многообразие внешнего мира. Эту особенность можно использовать для изучения иноязычной терминологии, выбрав за отправную точку текст на родном языке.

А дальше что?

«Ну, написал я статью, а дальше что?» А дальше можно совершенствовать статью – прямая дорога на страничку «Портал:Качество». Имеет смысл обратить внимание на две скромные ссылки: «Википедия:Хорошие статьи» и «Википедия:Избранные статьи».

Статья – хорошо, а хорошая статья – ещё лучше. Поэтому идите на страничку «Википедия:Кандидаты в хорошие статьи», изучайте требования, доводите статью до нужной кондиции и оставляйте запрос на страничке кандидатов на присвоение высокого звания. В саму статью в таком случае следует добавить шаблон {{Кандидат в хорошие статьи}}. Если статью сильно ругать не будут и как минимум три участника её похвалят, то она становится хорошей.

Следующая цель – сделать статью избранной. Совершенство не имеет границ. За активное участие в деле развития Википедии можно даже орден получить («Википедия:Ордена») – дел и целей непочатый край.

Википроекты

В реальности, информация хранится не только в виде энциклопедических статей. Поэтому от того же корня, от которого произошла Википедии отпочковались и другие Вики-проекты.

Викитека (http://ru.wikisource.org/wiki/) – архив текстов. Множество документов не нуждается в развитии. В исторических текстах, даже ошибки подвергаются анализу, но уж никак не исправлению. Документы, отданные на хранение в Викитеку, не предполагают больших правок.

Викисклад (http://commons.wikimedia.org/wiki/) – глобальное хранилище изображений, звука, видео и тому подобное. Хранилище глобальное, поэтому нет разделения по языковому принципу.

Викицитатник (http://ru.wikiquote.org/wiki/) – подборка цитат. Цитаты с указанием источника можно использовать без разрешения автора.

Викисловарь (http://ru.wiktionary.org/wiki/) – словарь. Не просто какой-то там словарь, а словарь грамматический, толковый, фразеологический, этимологический, ну и, естественно, многоязычный. Пока в словаре всего 10426 статей – очень молодой проект.

Викиучебник (http://ru.wikibooks.org/wiki/) – по идее, должно быть хранилище свободно развиваемых учебников. На сегодня это скорее небольшой сборник статей с не совсем компетентным в написании учебников администратором.

Викивиды (http://species.wikimedia.org/wiki/) – биологам посвящается. Каждое живое существо должно иметь свою страничку. На общем сайте есть и русскоязычный портал.

Викиновости (http://ru.wikinews.org/wiki/) – новостная лента, коих сейчас довольно много, зато на викидвижке.

На мой взгляд, Википедия слишком зациклена на простом тексте. Кроме Wiki-разметки, существуют и другие методы создания структурированных документов. У того же http://xxx.lanl.gov вполне можно и поучиться. Да и на PNG/JPEG/SVG свет клином не сошёлся. Не хватает хранилища для «контента», который требуется обрабатывать за пределами браузеров – для того же PDF, например, да и как публиковать в терминах Вики программы, не совсем понятно. Ждём проектов типа Викиархива и Википрограммы.

Легко создаваемый, можно сказать, естественный гипертекст уже вошёл в жизнь. «Плоские» текстовые разметки пока ещё живут очень даже неплохо, но чтобы жить дальше, им необходимо тоже взять на вооружение и эту идею.

Но идея естественного гипертекста была бы ничто без локомотива под названием Википедия. Локомотива, который можно скопировать на свой диск («Википедия:Как сделать копию Википедии») дабы собрать свою версию «паровоза». Это не наука – это технология. Именно этим она и замечательна. LXF



Учебники

Наши эксперты помогут вам с любым приложением Linux



ЕВГЕНИЙ БАЛДИН
Начинал с Агатов. Когда-то даже знал, что такое Робик.

Конфликт поколений

Классическая проблема отцов и детей: проблемы «отцов» «детям» абсолютно «до лампочки». Растет новое племя: молодое и незнакомое, для которых священные войны Gnome против KDE являются почти смыслом жизни. Те, кто чуть постарше (лет на пять-шесть), обзываются и ехидничают. Но разве их слушают? Разве они слушали кого-то в прошлом тысячелетии, когда образовывали свое техническое сообщество?

«Пакет IKARUS оказался слишком дорогой и сложной системой для персональных компьютеров, слишком профессиональной для дизайнеров нового поколения...»

Владимир Ефимов в предисловии к книге П. Карова «Шрифтовые технологии».

Опыт приходит только с опытом. Ценность образования тем выше, чем сложнее конкурсные испытания. Причем структура испытания фактически неважна – основное это сложность. Скомпилировал ядро, прочитал man-страничку, поставил другу Slackware – прошел тест, даже не на знания – на усидчивость и дотошность. Сейчас все стало сильно проще – ценность образования падает. Любая домохозяйка может поставить Ubuntu на свой персональный компьютер – образование становится общедоступным. Появляются личности, которые даже man-ов не читают, им XML подавай вместе с подсветкой – «халявщики» сбиваются в стайки.

Похоже, это закон развития любых полезных технологий: запускаются они высококвалифицированными профессионалами, но затем приходит толпа. Любопытная и нетерпеливая, как ребенок – тащит в рот всякую блестящую гадость. Чтобы движение продолжалось, эту толпу надо образовывать. То есть создавать ей сложности, после преодоления которых вырастет новое поколение профессионалов. Не надо обзываться и ехидничать – нужно обустроить полосу препятствий. E.M.Baldin@inp.nsk.su

В этом выпуске...

Совет месяца: Обратный SSH

Мы не станем извиняться за изобилие советов по SSH – это действительно один из наиболее универсальных инструментов в вашей Linux-системе. Многие люди используют его для ежедневного подключения к серверам, стоящим во всех уголках Земли. Мы уже обсуждали создание туннелей, использование SSH в качестве прокси-сервера, а месяц назад говорили об escape-последовательностях. Все эти советы имели одну общую черту – сервер посылал данные клиенту

Как вы уже, наверное, догадались, ситуацию можно обратить. Это бывает полезно для обхода брандмауэров, а также доступа к сети предприятия из дома без VPN. С «той» стороны, вам потребуется набрать следующее:

```
ssh -R 1234:localhost:22 home_machine
```

Естественно, **home_machine** следует заменить на IP-адрес вашей машины в сети. Для проброса соединения мы используем порт 1234, так что он должен быть доступен и не заблокирован межсетевым экраном. Создав соединение в офисе, отправляйтесь домой и набирайте:

```
ssh workusername@localhost -p 1234
```

Вы подключитесь к вашей корпоративной машине и сможете работать не хуже, чем в офисе.

Процедуру несложно модифицировать для доступа к файл-серверам и даже удаленному рабочему столу через VNC. Единственная проблема, с которой вы можете столкнуться – таймаут SSH-соединения. Чтобы решить ее, отредактируйте файл **/etc/ssh/sshd.conf** и укажите в нем опции 'KeepAlive yes' и 'ServerAliveInterval 60', чтобы соединение не обрывалось автоматически.

QuiteInsane:

Не только большой бизнес может извлечь доход из старого оборудования: **Энди Ченнел** намерен реабилитировать скромный сканер.





Сейчас, когда видеокамер понавстраивали чуть ли не в каждый утюг, сканеру угрожает забвение. И это позор, ибо сканер – весьма даровитое устройство, и в Linux есть для него немало отличных программ. На данном уроке мы исследуем приложение под названием *QuiteInsane* – это пакет на базе *Qt*/KDE, но не столь привязанный к рабочему столу, как, скажем, *Kooka* или GnomeScan. Мы также считаем, что он немного дружелюбнее к пользователю и, главное, прекрасно интегрируется как расширение в *Gimp 2.0*, то есть позволяет сканировать прямо в *Gimp*.

Используем это приложение для сканирования прошлых выпусков *Linux Format*, чтобы сослать физический журнал на чердак. Применим также систему распознавания текста (OCR) для перевода полученных изображений страниц в простой текстовый формат, неоценимый для хранения содержимого документов благодаря мизерности требований к памяти, и вдобавок допускающий редактирование; система пригодится и для извлечения текста из печатных источников – его можно потом добавлять в новые электронные документы, например, web-страницы.



Наш эксперт

Энди Ченнел
Энди делает свои первые шаги в Linux уже шесть лет, а технологиями интересуется еще со времен Dragon 32.

Часть 1 Установка QuiteInsane

Как и многие другие приложения Linux, *QuiteInsane* – не более чем графическая оболочка консольного приложения, в данном случае – библиотек сканирования *Sane* и пакета распознавания символов *GOCR*. По старой доброй традиции «из вывески все ясно» *Sane* [по-английски, «здравый», – прим. пер.] расшифровывается как Scanner Access Now Easy (Доступ к сканеру отныне прост), и эта библиотека лежит в основе большинства Linux-приложений для сканирования. Поэтому возможности *QuiteInsane* [по-английски, «просто безумно», – прим. пер.] подобны имеющимся в других аналогичных приложениях.

Рай для оборудования

Большинство изготовителей сканеров перешли на стандартный USB-интерфейс, отказавшись от параллельного или SCSI, так что дни мучений с драйверами практически позади. В базе поддерживаемых *Sane* устройств, например, имеется 70 сканеров только от Epson, и все, кроме четырех, отмечены как имеющие «хорошую» или «полную» поддержку. Среди исключений – самые последние модели Stylus (CX-5800 и StylusScan 2000), но похоже, что скоро и они будут поддерживаться. HP поживает несколько хуже, но и тут большинство устройств отмечены как поддерживаемые «хорошо» или «полностью».

Лучший способ получить поддерживаемый сканер – это выяснить все детали, а потом уж идти в магазин. Наиболее полный список поддерживаемых устройств находится на www.sane-project.org/sane-mfgs.html. Он постоянно обновляется, и в настоящее время содержит данные о 1273 сканерах, из которых 348 «хорошистов».

Если вы используете ОС на основе Debian, например, Linspire, Xandros или Ubuntu (или сам Debian), то пакет *QuiteInsane* должен быть доступен по методу *apt-get*/*Synaptic*: достаточно открыть терминал и набрать:

```
sudo apt-get install quiteinsane
```

Введите пароль root и подивитесь на колдовство менеджера пакетов Debian. Если вы собираетесь использовать приложение из-под *Gimp*, повторите эти же манипуляции с пакетом **gimp2.0-quiteInsane**. Если вы пользуетесь *Synaptic*, следует найти указанные пакеты, а затем установить обычным способом. Если же вы предпочитаете RPM, то найдете подходящий пакет или при помощи стандартного менеджера пакетов (Yum, *YaST* и так далее) или посредством http://rpmfind.net или http://rpm.pbone.net. А те, кто готов расправить обретенные крылья, конечно же, могут выбрать сборку из исходных текстов.

На нескольких системах, которые я опробовал, запись для *QuiteInsane* почему-то не появлялась в меню; в таком случае вы можете запустить приложение, выбрав пункт Run Command [Выполнить команду] в меню K/Gnome и набрав **quiteInsane**. Можно также создать ярлык для этого приложения на рабочем столе или в меню – щелкните правой кнопкой мыши на рабочем столе, выберите Create New > Link To Application [Создать > Ссылка на приложение], задайте имя и добавьте '*QuiteInsane*' в строку Command [Команда].

Месяц назад Мы запускали Windows-приложения под Linux, через *Wine*.

посканируем!

Часть 2 Введение в интерфейс QuiteInsane

Интерфейс *QuiteInsane* имеет много настроек, и поддерживает несколько режимов пользовательского интерфейса, включая вкладки, множественные окна и списки. На рисунке справа выбран режим отображения List [Список]: он выглядит наиболее «здравым» и отображает на экране больше информации, чем другие.

[В настройки включен даже модуль перевода элементов интерфейса, причем файл с переводом выбирается через стандартный диалог, так что любой пользователь может самостоятельно локализовать *QuiteInsane* под себя. Но учтите: приложение работает ТОЛЬКО при подключенном сканере. – прим. пер.]

1 Опции сканирования [Scan Options]

Этот раздел предоставляет доступ к различным настройкам сканера. То, что вы здесь выберете, повлияет на весь пользовательский интерфейс, в частности, панели. Большинство операций прекрасно выполняются без перехода в Scan Mode [Режим Сканирования], но мы также кратко опишем другие доступные опции. Вкладка Advanced [Дополнительно] содержит опции для настройки скорости сканирования и глубины цвета. Производимые здесь изменения могут занять много времени и требуют множества тестовых сканирований. Это верно и для диалога Colour Correction [Коррекция цвета], позволяющего настроить баланс каналов RGB (обратитесь к нему, если ваш сканер постоянно искажает цвета). Набор опций вкладки Preview [Предварительный просмотр] для настройки предварительного сканирования зависит от вашего устройства; Geometry [Геометрия, Размещение] содержит предопределенные размеры изображений; и, наконец, вкладка Optional Equipment [Дополнительные устройства] будет пуста, если ваш сканер не предусматривает автоподачи или слайд-адаптера.

2 Режим Сканирования [Scan Mode]

Наш выбор здесь зависит от требуемого продукта. Сюда входят Scan Mode [Режим сканирования], Gamma Correction [Коррекция гаммы] и Resolution [Разрешение]. Режимов может быть по крайней мере три. Binary [Двоичный, Черно-белый] представит каждый пиксел результата белым либо черным; его можно использовать для работы с фотографиями, но разработан он был для сканирования и последующего распознавания букв. Grey [Оттенки серого] создает изображение в оттенках серого, а Colour [Цветной] – в полном 24-битном цвете. Гамма-коррекция пригодится в случае, если ваш сканер выдает темные изображения: чем больше значение, тем ярче результат. И, наконец, разрешение – это количество пикселей на единицу длины. Наихудшее качество, 75 dpi [dot per inch, пикселей на дюйм], годится для изображений, публикуемых в Интернете, а наилучшее, 600 dpi – для распечатки фотографий. С подъемом по этой шкале размер получаемого файла стремительно растет.

3 Статистика [Statistics]

Окно статистики сообщает, насколько большим будет изображение, как по количеству пикселей, так и по ожидаемому размеру файла. Но это только прикидка: если вы сохраните изображение в сжатом формате (типа JPEG или PNG), файл, скорее всего, будет поменьше.

4 Тип изображения [Image Type]

Этот выпадающий список определяет некоторые стандартные опции, а также то, что мы собираемся делать с результатом. Опция Temporary/Internal viewer [Временный/встроенный просмотрщик] пригодится, если нужно вырезать кусок и вставить в новый документ; Single File [отдельный файл] сканирует и записывает результат в отдельный документ; OCR [Оптическое распознавание символов] оптимизировано для преобразования графики в текст, Copy/Print [Копировать/Печатать] шлет результат прямиком на принтер; Multiscan [Множественное сканирование] позволяет установить количество сканирований и может сочетаться с автоподачей листов или слайд-адаптером; а Save [Сохранить] сканирует сразу в файл.



5 Параметры настройки [Configuration options]

Здесь можно найти опции для настройки интерфейса пользователя, открыть панель Предпросмотра, настроить ваше устройство, начать сканирование и использовать встроенный просмотрщик изображений.

6 Предпросмотр [Start preview]

Эта кнопка запускает предварительное сканирование: на его основании можно более точно выбрать участок для финального сканирования.

7 Параметры предпросмотра [Preview options]

Эти кнопки используются для увеличения и выбора частей изображения. Первая иконка увеличит выделенную область. Две иконки в середине – отмена и повтор действий, а четвертая – сброс настроек в предпросмотр полного изображения.

8 Параметры выделения [Selection options]

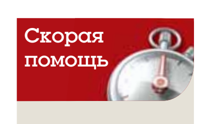
Инструменты для автоматического выбора области сканирования; работают на основе цвета, так что цветную фотографию посреди большого белого поля приложение выделит, но более беспорядочный фон может вызвать проблемы.

»



Старайтесь выбирать разрешение в соответствии с работой – для использования в web достаточно 75 dpi.

Часть 3 Сканирование



Скорая помощь

Можно задать размер отсканированного изображения, например, потребовать, чтобы оно умещалось на CD, а настройки геометрии тогда будут подобраны автоматически.

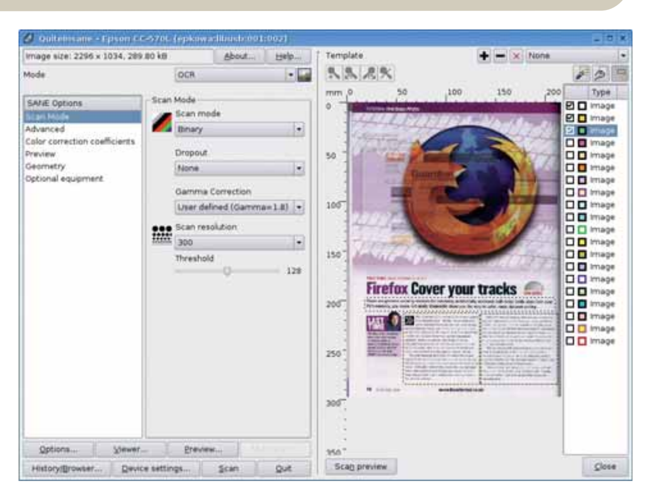
Поместив на стекло сканера документ, первое, что вы должны сделать – получить предварительное изображение. Нажмите кнопку Preview [Предпросмотр]: будет выполнено быстрое сканирование, результат которого отобразится в правой части окна. Теперь используйте или прямоугольник выделения (со стандартной пунктирной рамкой) для захвата всего изображения, или мышь, чтобы вырезать его часть. Я сканирую изображение для web-сайта, поэтому выбираю Цветной режим сканирования и разрешение 75 dpi. Размер файла увеличивается с разрешением сканирования – полная журнальная страница формата A4 при 75 dpi потребует несколько сотен килобайт, но то же изображение при 600 dpi займет почти 50 МБ. Просто для сравнения: это почти размер целого дистрибутива Damn Small Linux. Зато когда в дело вступает сжатие в формат JPEG, файл становится более подъемным. Размер файла также зависит от используемого режима: так, черно-белое изображение будет намного меньше, чем в оттенках серого, которое в свою очередь значительно меньше того же в цвете.

Если вам необходима большая точность выбора выделенной области, нажмите первую иконку выделения (*см. аннотацию 8 на предыдущей странице*), и эта область увеличится. Подогнав рамку, нажмите кнопку Scan [Сканировать] – начнется сканирование. По его завершении – в зависимости от устройства и настроек, может пройти несколько секунд или минут – откроется стандартное окно выбора файлов, и вы сможете сохранить изображение в выбранном вами формате и месте. Наилучший результат дает формат TIFF, сжимающий без потерь качества, но для более эффективного использования дисковой памяти выберите JPEG или PNG.

Итак, наше первое сканирование завершено; теперь можете открыть и обработать картинку в *Gimp* или в другом редакторе изображений.

OCR: оптическое распознавание символов

Далее попробуем отсканировать какой-нибудь текст из журнала, как я указывал во введении. Данное приложение, как и многие современные программы для сканирования, не ограничивается обработкой текста как одного длинного потока, вроде письма, а имеет инструменты для определения и захвата в должном порядке элементов достаточно сложной верстки. Во-первых, выберите режим OCR, используя выпадающий список меню Mode [Режим]. Скорее всего мы получим наилучший результат, выбрав опцию Binary [Черно-белое] в Scan Mode [Режим сканирования]; затем используем инструмент выделения и



› *QuiteInsane* **отлично работает с блоками текста: каждая рамка для обозначения их границ имеет свой цвет.**

последующего увеличения текстового раздела. Увеличьте также разрешение до 300 dpi, обычно это дает наилучший результат.

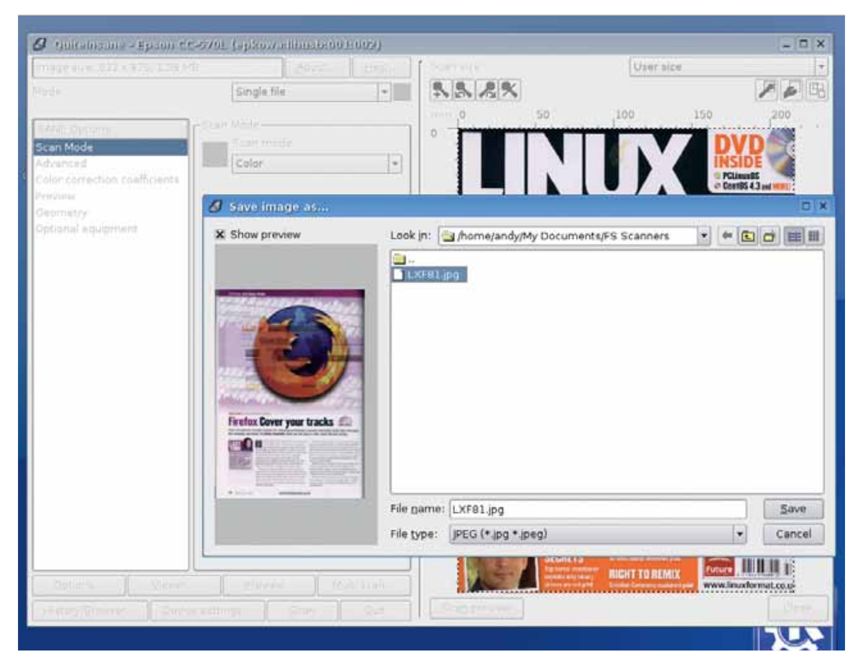
Выберите область Image Type [Тип изображения] – откроется дополнительная панель с множеством цветных квадратиков по правому краю (*см. рис. выше*). Если вы щелкнете на первом (белом) квадратике и убедитесь, что рядом с ним появилась пометка, то в окне предпросмотра появится новая прямоугольная рамка. Окружите ею тот кусок текста, с которым хотите работать. Затем выберите второй цвет в списке (в моей версии, оранжевый) для создания другой рамки выбора; выберите второй кусок текста. Продолжайте делать это до тех пор, пока не будет выбран весь нужный текст, в правильном порядке: например, колонки журнальной статьи выбираются слева направо. Вы можете переразместить эти элементы, указав соответствующий цвет в списке и отрегулировав маркеры выделения. Выбрав все, что нужно, нажмите кнопку Scan. Каждое выделение будет отсканировано и обработано OCR-приложением, и в нашем случае результатом будут три документа, представленные на экране встроенного в *QuiteInsane* текстового редактора.

Распознавание символов – процесс неточный, и финальный текст может потребовать небольшого [а в случае текста на русском языке и большого, – прим.ред.] редактирования. В моем примере приложение слегка запуталось со смесью курсива и прямого текста в статье, и случайно проявились яркие цвета обратной стороны сканируемой страницы. Подобно другим OCR-приложениям, наше помещает на концах всех строк символ жесткого перевода строки. Результат, тем не менее, чего-то да стоит: пусть не обошлось без подправки, это все равно быстрее, чем перенабирать текст вручную.

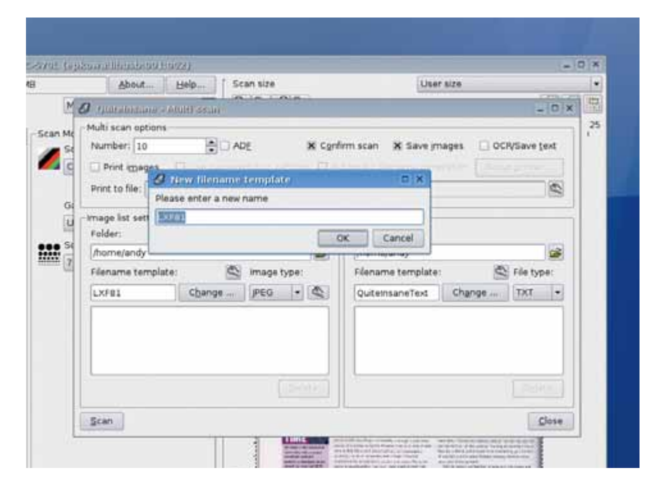
Сканирование для архива

Теперь подымем планку: отсканируем серию документов в папку на жестком диске, и дадим им всем одинаковое имя, сопровождаемое личным номером. Если у вас есть автоподача, то многое выполнится автоматически, но у нас ее нет, и мы произведем настройку немного по-другому. Во-первых, установим режим сканирования и разрешение для архива; я выбрал Цветной и 75 dpi, потому что изображение будет только просматриваться на экране, но не печататься. Все остальное не меняем, и настраиваем рамку выбора для захвата всего стола (сканируемой области).

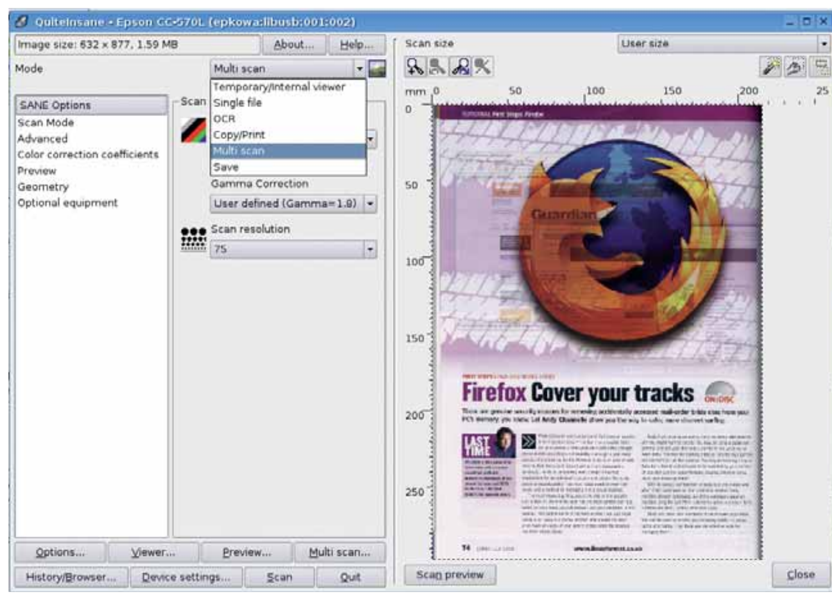
Выберем в выпадающем списке режимов Mode множественное сканирование – Multiscan; откроется диалог настройки множественного сканирования (*см. рис. вверху слева на следующей странице*), мы пойдем по нему сверху вниз. Сначала выберите желаемое число сканирований и, если у вас есть автоподача, поставьте отметку ADF. Например, для архивирования 128-страничного журнала потребуется 128 скани-



› **Завершив сканирование, вы можете сохранить результат на диске.**



› Установите между сканированиями паузу, чтобы вы смогли перевернуть страницу.



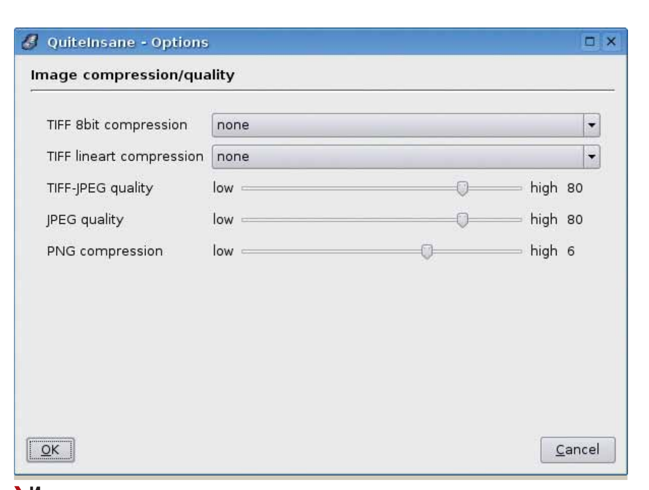
› В режиме Multiscan сканируется набор страниц; показан диалог этого режима.

рований. Если автоподача у вас есть, но опцию ADF вы не пометили, приложение не будет знать, что ее надо использовать. Далее нужно выбрать Confirm Scan [Подтверждение сканирования] – это означает, что приложение будет делать паузу после каждого сканирования, чтобы мы могли перевернуть страницу журнала, а потом уж продолжать – и Save Scan [Сохранить сканируемое] для сохранения файла на диске. Вы можете сканировать и в оперативную память, но не очень понятно, зачем это может быть нужно.

Вы можете нажать кнопку Print [Печать], и каждое из полученных изображений будет сохранено и отправлено на принтер – получим аналог копировальной машины. Эта опция весьма удобна, если ваш принтер лазерный, но владельцы струйного принтера скоро обнаружат, что очередь печати растет, поскольку сканер работает где-то раз в 50 быстрее, чем такой принтер.

Упорядочите ваши новые данные

Теперь нужно настроить структуру файлов в Image List Settings [Настройки списка изображений]. Скорее всего, большинство опций будут вам знакомы. Например, можно использовать стандартное окно выбора файлов для выбора типа изображения, или указать место для файлов в строке Folder [Каталог]. Добавьте имя в строку Filename Template [шаблон имени файла], оно будет первой частью имени файлов. Где бы в этом приложении вы ни увидели иконку с гаечным ключом, это всегда указание на дополнительные опции, так что щелкайте на той, что стоит рядом с шаблоном имени файла, для определения оставшейся части имени, которая будет генерироваться при каждом сканировании.

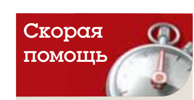


› Иконка с гаечным ключом вызывает настраиваемые опции вроде этой.

Здесь мы настраиваем приращение для номеров сканируемых изображений. Поскольку мы собираемся дойти до десяти и хотим сохранить упорядочение, отметьте опцию Prepend Zeros [Предварять нулями], а затем установите 2 в Counter Width [Разрядность счетчика]. Это означает, что первое изображение будет помечено 01, второе 02, и так далее. Если мы сканируем более 100 документов, то для правильного размещения файлов разрядность счетчика следует установить равной 3. В результате получится серия файлов с именами по типу **filename 01.jpg**, **filename 02.jpg** и так далее.

Теперь щелкните на иконке гаечного ключа под заголовком Image Type [Тип изображения] и укажите опции JPEG, например, качество (чем меньше значение, тем меньше файл и хуже качество изображения). Нажмите OK: к сканированию все готово; нажмите Scan, и первая страница будет отсканирована; вам предоставится возможность подтвердить ваше желание продолжать, что вы и сделаете, перевернув страницу. В результате получаем десять отсканированных и сохраненных страниц за пару минут. Это удобно, когда не нужно выделять отдельные части страницы (что требует индивидуального подхода) – но можно впоследствии перейти в *Gimp* и ликвидировать ненужные куски.

Пусть цифровые камеры привлекательнее, умнее и быстрее, но при помощи сканера вы всегда (по крайней мере, на данный момент) получите лучший результат при попытке распознать текст или сохранить страницы журнала или книги. Так что не давайте вашему сканеру расслабиться: заставьте его поработать! LXF



Скорая помощь

Законы об авторском праве в каждой стране свои, но в Великобритании вы не можете воспроизводить материалы, защищенные авторским правом, без согласия правообладателя, кроме как для некоммерческих исследований или частного изучения.

Встроимся в Gimp

Расширение *Gimp* для *QuiteInsane* – пример интеграции двух технологий. Как только вы установите расширение и запустите его из-под *Gimp*, используя File > Acquire > QuiteInsane > Scan, дальнейшие действия будут такими же, как и при отдельном запуске приложения. Отличие только в том, что после того, как вы все настроите и выполните сканирование, изображение автоматически загрузится в *Gimp* для редактирования или дальнейшей обработки, а *QuiteInsane* закроется.

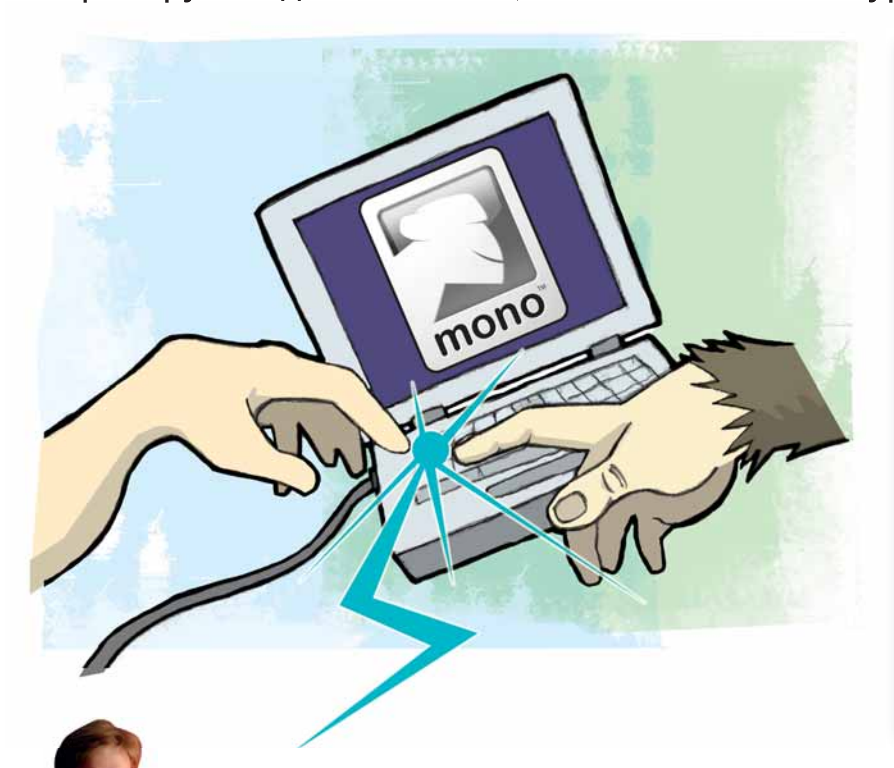


› Можно сканировать прямо в *Gimp*, благодаря его архитектуре расширений.

» **Через месяц** Мы научимся управлять пакетами с помощью APT.

Mono: Написать

Хотите научиться программировать? Имеете шанс: **Пол Хадсон** начинает новую серию руководств по Mono, C# и .NET. На этом уроке: первая программка.



Пакеты, нужные для установки Mono

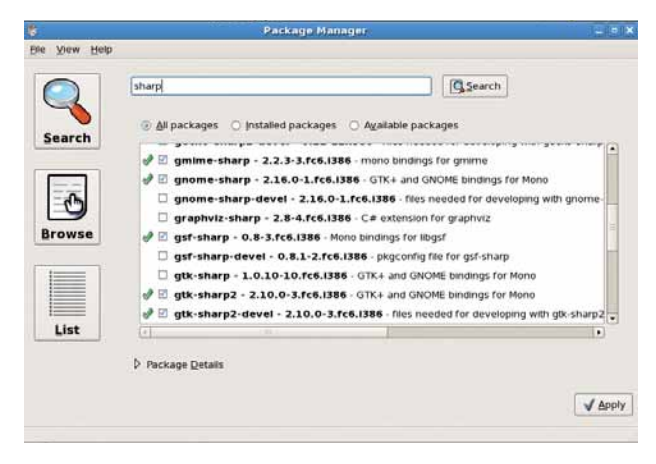
Вам кажется, что список чересчур велик для того, чтобы написать всего-навсего программу Hello World? На самом деле мы будем пользоваться этим набором пакетов на протяжении всей серии, создавая игры и мультимедиа-приложения, обрабатывая XML, считывая и сохраняя файлы, и делая многое другое.

- **avahi-sharp**
- **dbus-sharp**
- **evolution-sharp**
- **gecko-sharp2**
- **gmime-sharp**
- **gnome-sharp**
- **gsf-sharp**
- **gtk-sharp2**
- **gtk-sharp2-devel**
- **gtk-sharp2-doc**
- **gtksourceview-sharp**
- **gtksourceview-sharp-devel**
- **ipod-sharp**
- **mono-core**
- **mono-data**
- **mono-data-sqlite**
- **mono-debugger**
- **mono-devel**
- **monodevelop**
- **monodoc**
- **mono-extras**
- **mono-jscript**
- **mono-locale-extras**
- **mono-nunit**
- **mono-web**
- **mono-winforms**

Наш эксперт

Пол Хадсон полагает, что Mono – лучшая вещь со времен мультфильма *Pinky and the Brain*, и сейчас поддерживает два проекта на основе Mono на SourceForge.

На программировании держится мой мир, и я хочу, чтобы оно поддержало и ваш. Если вы сроду не программировали, то эта серия для вас: я попытаюсь рассказать, как стать профессиональным программистом, и сделаю все, чтобы этот процесс был увлекательным и интересным. Мы будем использовать Mono – открытую программную платформу, совместимую с Microsoft .NET. Если вы никогда о таком не слышали, ничего страшного – мы рассмотрим теорию позже.



› Пакеты устанавливаются через диалоговое окно Fedora Добавить/Удалить Программы. Нужный пакет быстро найдется Поиском по ключевому слову 'sharp'.

На этом уроке мы начнем с установки Mono, разрешим его зависимости и напишем программу Hello World: как вы, возможно, догадались, цель этого проекта – заставить Mono вывести на экран строку 'Hello World!'. В качестве базового дистрибутива для этого учебника я взял Fedora Core 6. Он включает все необходимое, чтобы научиться программировать в Mono, и если вы используете ту же версию, то можете в точности следовать моим инструкциям; если нет, то можете посмотреть врезку *В помощь Fedora-ненавистникам* на стр. 58.

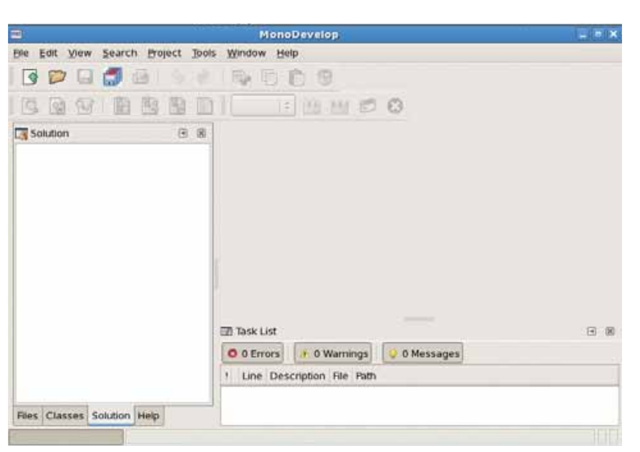
В FC6 большая часть работы уже проделана за вас, но чтобы получить от этого учебника все, необходимо добавить несколько дополнительных пакетов. Выберите пункт Add/Remove Software (Добавить/Удалить Программы) из меню Applications (Приложения), а когда появится окно, выберите режим просмотра в виде Списка (List view). Проверьте, что помечены все пакеты, указанные во врезке *Пакеты, нужные для установки Mono* (см. выше). Некоторые из них уже установлены, но большинство необходимо выбрать самому. Пометив их, нажмите кнопку Apply (Применить), чтобы Fedora скачала последние версии этих пакетов и установила их – на это может уйти несколько минут.

Итак, мы установили Mono, и теперь вы можете получить доступ к разным частям системы. Возможно, на наших уроках пригодится лишь небольшое подмножество этих пакетов, зато потом вы сможете приняться за задачи по своему вкусу. В список включен *Evolution-sharp* для доступа к почте и календарю *Evolution*; *gecko-sharp*, для встраивания в ваши приложения web-браузера *Gecko/Mozilla*; *ipod-sharp* для написания программ доступа к iPod.

Суффикс 'sharp' означает, что данная библиотека предназначена для .NET: например, пакет *foo-sharp* позволяет использовать с Mono библиотеку *foo*. 'Sharp' используется потому, что самый популярный

Hello, World!





› **MonoDevelop – предпочтительная IDE для Mono, но с ней не все просто. Мы раскроем ее секреты в последующих выпусках…**

язык программирования для .NET – это C#; музыканты читают C# как «си-диез», а по-английски это произносится «си-шарп».

Самый важный установленный пакет – *MonoDevelop*, главная среда разработки приложений в Mono. Программы можно набирать и в обычном текстовом редакторе, но зачем заниматься ерундой? В *MonoDevelop* множество полезных инструментов, которые сэкономят вашу энергию. Тем более, все эти пакеты уже скачаны и установлены, а Fedora добавила в меню специальный пункт запуска *MonoDevelop*: выберите Applications (Приложения), затем Programming (Программирование), и щелкните по ярлыку *MonoDevelop*.

Все об интеграции

MonoDevelop – интегрированная среда разработки (IDE) для программистов на Mono. Концепция IDE – включить в одно приложение необходимые инструменты, чтобы выполнять больше работы с минимумом трудностей. В *MonoDevelop* мы будем производить и кодирование, и компиляцию, и отладку – это мощнейшая среда разработки! Правда, оборотной стороной её «мощи» является «трудность в обучении» – на первый раз *MonoDevelop* выглядит немного ошеломляюще, но большую часть функций вы пока можете игнорировать. Я буду объяснять новые возможности по ходу их надобности.

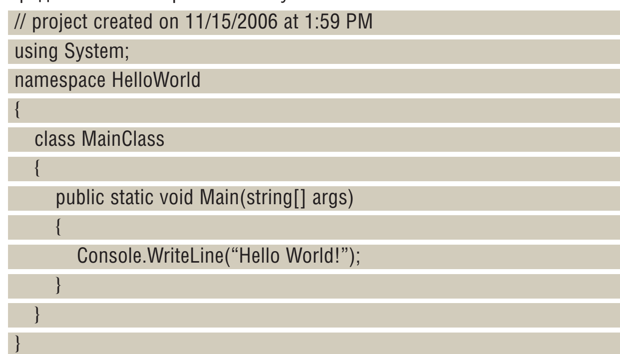
Займемся нашим первым проектом: Hello World. Это простая задача, и она даст нам прочную основу для дальнейшей разработки других программ.

Итак, в меню File выберите New Project. При появлении нового окна выберите в левой панели C#, затем Console Project на правой панели. В разделе Location (Местоположение) назовите проект HelloWorld. Поместите его в каталог, где будете сохранять свою работу – я обычно использую каталог **sandbox** [англ. «ящик с песком» – на таких проигрывают тактику военных операций, – прим. ред.], но вы вольны выбирать сами. Убедитесь, что не выбран пункт Create Separate Solution Subdirectory, и все: нажмите New, и *MonoDevelop* создаст скелет проекта.

Несколько секунд жужжания жесткого диска, и *MonoDevelop* оживает: слева вы увидите несколько файлов (включая **Main.cs** и **AssemblyInfo.cs**), а справа – код. Пока не обращайте на него внимания: нажмите F5. Для *MonoDevelop* это значит, что вы хотите запустить программу; он тут же компилирует код и запускает его.

Под панелью с кодом написано 'Application Output' (Вывод приложения), и после нажатия F5 на экране появится надпись 'Hello World!'. Миссия выполнена! Ну, не шибко-то радуйтесь: я немного слукавил. *MonoDevelop* поставляется с уже готовым проектом Hello World!, а на самом деле мы ничего не кодировали. Зато код проекта можно редактировать; вот и попробуем.

MonoDevelop сгенерировал 13 строчек кода, из которых только 5 представляют интерес. Вот что у нас есть:

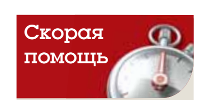


```
// project created on 11/15/2006 at 1:59 PM
using System;
namespace HelloWorld
{
   class MainClass
   {
      public static void Main(string[] args)
      {
         Console.WriteLine("Hello World!");
      }
   }
}
```

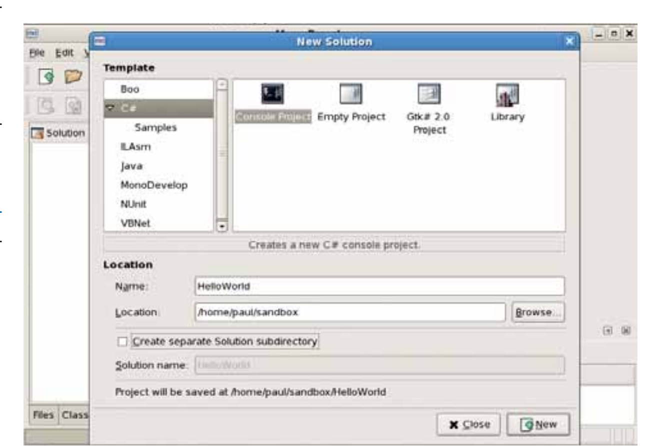
Разбираем программу

Первая строка начинается с двух слэшей (//), они означают, что строка является комментарием. Преобразуя исходный код в нечто исполнимое компьютером (именно это и называется «компиляцией»), Mono удаляет все комментарии. Напишите в программе хоть тысячу строк комментариев – они никак не повлияют на скорость ее исполнения, потому что компилятором просто игнорируются. Комментарии полезны лично нам: с их помощью мы поясняем, как работает программа или почему мы написали именно так. Этот комментарий *MonoDevelop* вставил автоматически, чтобы пометить, когда был создан проект – проку от него мало, поэтому можете удалить эту строку, если хотите.

Следующие две строки касаются первой части нашей теории: пространств имен. Платформа .NET велика. Очень велика! Здесь умещаются библиотеки для работы с XML, графикой, сетью, пользовательским интерфейсом, файлами, безопасностью… и пр. На свете есть тысячи библиотек разработчиков, которые вы можете скачать и включить в собственный код. Каждая из них дает своим компонентам свои собственные имена, поэтому если загрузить их одновременно, то возможности именования в вашей программе будут серьезно ограничены. C# »



Если вы хотите, чтобы MonoDevelop скомпилировал ваш код, но не запускал его (что понадобится, когда вы захотите задать своей программе аргументы), просто нажмите клавишу F8.



› **Для создания нового проекта MonoDevelop предлагает на выбор несколько полезных шаблонов.**

» решает эту проблему с помощью пространств имен: каждая библиотека находится где-то в иерархии .NET, куда вам необходимо явным образом загрузить ее.

System – это пространство имен, включающее базовые вещи .NET, в том числе возможность выводить на консоль сообщение 'Hello World!'. Не будь строки using System;, строка, начинающаяся с Console.WriteLine, не сработала бы: Mono не знал бы, где искать 'Console'. Нам пришлось бы явно указать, что мы имели в виду Console из библиотеки System, изменив строку на следующую: System.Console.WriteLine. Написав using System;, мы просто сокращаем работу по набору кода. Заметим, что строка кода заканчивается точкой с запятой: для Mono это признак окончания строки.

После строки с using мы определяем свое собственное пространство имен: HelloWorld. И снова, это гарантирует, что наш новый код не будет конфликтовать с наличным кодом Mono. Если имена в ваших проектах сугубо экзотические, без пространств имен можно обойтись, но все же рекомендую оставить эту строку.

Сразу после объявления пространства имен следует открывающая фигурная скобка ({). Вообще-то в коде полно { и }, поэтому вы, вероятно, хотите узнать, зачем они нужны. В C# фигурные скобки используются как метки начала и конца блока кода. В приведенном выше примере, открывающая фигурная скобка сразу после namespace HelloWorld говорит «пространство имен начинается здесь». В конце кода находится закрывающая скобка, означающая, что «пространство HelloWorld заканчивается здесь». Все, что находится между открывающей и закрывающей скобкой, является частью пространства имен HelloWorld.

В коде есть еще четыре открывающих и закрывающих скобки, каждая из которых помечает начало и конец соответствующего блока. Компилятору неважно, насколько аккуратно расставлены скобки, однако выравнивание положений открывающей и закрывающей скобок облегчает чтение – чтобы найти конец блока, достаточно проследовать по вертикали вниз. Заметим, что после скобок не нужна точка с запятой: скобки ничего не делают в коде, а просто представляют структуру.

C# реализует концепции объектно-ориентированного программирования, то есть в коде можно определить сущности (называемые классами), затем создать экземпляры этих сущностей (называемые объектами), и совершать над ними некие операции. Например, если вы пишете игру про гонки, вы создаёте класс Car (Машина), затем создаёте десять объектов этого класса и помещаете их на трассу. У каждой машины своя позиция и скорость – это данные объекта. Внутри нашего пространства имени находится один класс – *MonoDevelop* назвал его безликим именем MainClass: ведь неизвестно, что мы собрались запрограммировать.

MainClass не содержит собственных данных, но содержит метод Main. Методы – это действия, которые вы хотите заставить объект выполнять по вашей команде. Так, в примере с машиной вам понадобятся методы Accelerate [Ускориться], Decelerate [Снизить скорость], ChangeGear [сменить передачу] и Crash [сломаться]. (Имена не могут содержать пробелов, но допускают знаки подчеркивания). Эти методы заставляют объекты совершать действия – в случае с Accelerate произойдет увеличение скорости машины.

Метод Main выглядит довольно коряво, потому что предварен аж тремя ключевыми словами: public, static и void. Пока оставим их в покое. Кроме того, в скобках метода Main стоит string[] args, нагоняя еще больше страху. Но этим мы тоже займемся попозже. Надеюсь, вы заметили открывающую скобку на следующей строчке – она отмечает начало метода.

Последняя заслуживающая внимания строка – Console.WriteLine. Console – это объект, который представляет **консоль**, в которой работает наша программа и куда направится весь вывод. WriteLine – метод этого объекта, выводящий текст на консоль. Чтобы вызвать WriteLine («вызвать» – то есть исполнить код), мы просто передаем ему строку текста, заключенную в кавычки, например так:

```
Console.WriteLine("Hello World!");
```

Скобки – (и) – используются, чтобы сообщить Mono, что мы вызываем метод WriteLine; без них он не будет знать, что делать.

Всего 13 строк кода, а сколько объяснений! Зато вы уже впитали довольно много теории.

Изучаем возможности WriteLine()

Текст 'Hello World!' не особо нов, но внутри кавычек можно поместить любую другую строку для вывода (не забывайте нажимать F5, чтобы MonoDevelop перекомпилировал и выполнил ваш код). Любой-то любой, но есть одно исключение, если в тексте имеются кавычки:

```
Console.WriteLine("Затем Эффи сказал "Я зануда", и вправду так и думал.");
```

Попытавшись набрать этот текст в *MonoDevelop*, вы увидите, что *MonoDevelop* выделяет текст красным цветом. Точнее, вы увидите, что Затем Эффи сказал выделено красным цветом, затем **Я зануда** – черное, а затем , и вправду так и думал. выделено опять красным. Подумайте: если в Mono кавычки помечают начало и конец текста, то как он сможет уловить разницу между кавычкой, которую вы хотите вывести, и кавычкой, означающей конец текста? Да никак. Поэтому он сочтет, что выводимая строка текста –'Затем Эффи сказал', а следующая часть, 'Я зануда', – уже код C#. Mono это не понравится.

Если вы хотите включить кавычки в ваш текст, предупредите Mono, что кавычку надо вывести, а не считать её концом текста. В C# это можно сделать с помощью специального модификатора [escape character]: \. Модификатор сообщает C#, что следующий за ним символ интерпретируется особым образом. Если вы хотите вывести кавычки, необходимо поместить перед кавычкой модификатор:

```
Console.WriteLine("Затем Эффи сказал, \"Я зануда!\", и вправду так и думал.");
```

Другими модификаторами являются \n, означающий «начало следующей строки»; и \\, означающий «вывод обратного слэша» (то есть знак \ выводится, а не считается модификатором). Вообще-то модифи-

Ориентирование объекта

Объектно-ориентированное программирование (ООП) – весьма непростая штука, но в конечном итоге оно ловко задумано, потому что его цель – сделать ваши объекты такими умными, что плохой код для них и написать трудно. Мы рассмотрим концепции ООП более подробно на протяжении следующих выпусков, однако вы уже выучили самую сложную часть: разницу между классами и объектами. Вы же помните, в чем разница, правда?

Скорая помощь

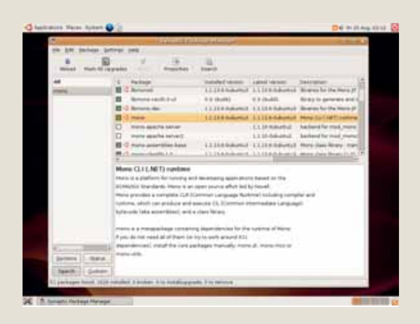
Панель слева в MonoDevelop отображает ваши файлы, а внизу панели вы можете видеть вкладку Solution. Если вы перейдете на вкладку Help, то сможете просматривать документацию Mono во время написания программы.

В помощь Fedora-ненавистникам

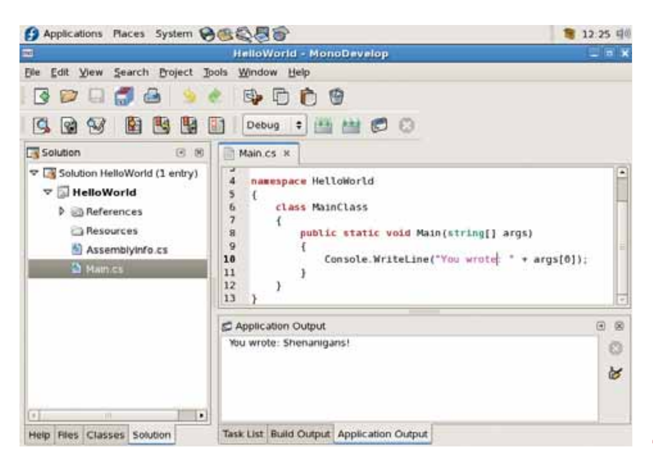
Ну, может, вы вовсе и не ненавидите Fedora – я уверен, есть много важных причин выбрать Ubuntu/SUSE/Mandriva/Gentoo/MikeOS. Однако все шаги этого руководства тестировались на Fedora Core 6; если вы используете другую систему, я не гарантирую, что все здесь описанное будет работать.

Большинство дистрибутивов включают Mono, или по крайней мере позволяют его поставить через менеджер пакетов. Имена пакетов в них очень похожи, хотя Debian/Ubuntu предпочитают как часть имени для пакетов разработки использовать суффикс **-dev** вместо **-devel**.

FC6 поставляется вместе с Mono 1.1.17 и MonoDevelop 0.12. Если у вас старые версии, и в них есть проблемы, можете скачать Linux Installer for x86 binary с www.mono-project.com/Downloads. Установщик скопирует файлы в каталог на вашей системе; если вы установите его как основную рабочую среду, то сможете следовать этому руководству. И, пожалуйста, не стесняйтесь задавать вопросы на форуме Linux Format по адресу www.linuxforum.ru.



› **Если у вас Ubuntu, осторожнее: Dapper-версия Mono может быть несовместима с нашей.**



› Законченный продукт: ввод пользователя передается в метод WriteLine(), и его сообщение отображается на экране.

катор \n не так уж и нужен, потому что метод WriteLine автоматически добавляет символ новой строки в конец каждой выводимой строки. Например, если бы вы изменили код на следующий:

```
Console.WriteLine("Фонарный столб он кулаком ");
Console.WriteLine("Ударил что есть сил,");
Console.WriteLine("И гордо заявил потом,");
Console.WriteLine("Что призрака сразил.");
```

то на каждой строке напечаталось бы по одной строке текста. Управлять переносом строки самостоятельно позволяет метод Console.Write, например, так:

```
Console.WriteLine("Фонарный столб он кулаком "\n);
Console.WriteLine("Ударил что есть сил,"\n);
Console.WriteLine("И гордо заявил потом,"\n);
Console.WriteLine("Что призрака сразил."\n);
```

Параметр номер 0

Взгляните на часть string args[] в методе Main на стр. 57. Каждое определение метода должно заканчиваться открывающей и закрывающей круглой скобкой. Между этими скобками мы можем определить, какие данные – если надо – мы будем передавать методу. Вспомним пример с машиной: как метод ChangeGear узнает, на какую передачу надо переключиться? Ответ очевиден: указать ему номер передачи. Вот это и есть параметр. Мы определяем параметры, которые должен получать наш метод, а C# проверяет, что при вызове метода использованы правильные данные. У каждого параметра есть тип и имя, по которому к нему будет происходить обращение в методе.

В случае с string args[] наш код утверждает три вещи: методу Main будет передан строго один параметр; доступ к нему из метода Main будет осуществлен через имя args; это будет массив строк (массив обозначается символом [], открывающая и закрывающая квадратные скобки). Строкой называется просто текст: любая последовательность символов, начинающаяся и заканчивающаяся символом двойной кавычки. То есть "Hello World!" – это строка. Массив – это группа объектов одного типа, а значит, параметр args может быть пустым, может содержать одну строку или 100 строк – все зависит того, что передается в метод при его вызове.

Если вы программируете впервые, то, возможно, не поняли того, что я сказал, поэтому я очерчу проблему: где мы в действительности вызываем метод Main? Ответ: мы его НЕ вызываем! Однако же 'Hello World!' выводится на экран? Дело в том, что метод 'Main' – особый метод. Когда запускается наша программа, Mono автоматически ищет и вызывает метод Main, а также передает ему параметры командной строки. Мы привыкли набирать **cd /usr/bin**, чтобы сменить каталог; **/usr/bin** является параметром команды *cd*. Когда Mono вызывает метод Main, он посылает эти параметры в виде массива строк args, а это значит, что мы можем сделать нашу первую полезную программу: ввод данных пользователем для их дальнейшего применения.

Создайте новый проект (выберите File > New Project > C# > Console Project и уберите там дурацкую галочку) и назовите его "ParamPrint". Программа будет принимать один входной параметр командной строки и выводить его вместе со строкой 'Hello World!'.

В новом коде, который сгенерирует *MonoDevelop*, измените Console.WriteLine на следующую строку:

```
Console.WriteLine("Hello World: " + args[0]);
```

Так как переменная args – это массив строк, то необходимо сообщить C#, который элемент массива мы хотим вывести. Элементы массива нумеруются, начиная с 0, который обозначает первый элемент, 1 – второй элемент и т.д.; чтобы напечатать первый переданный параметр, необходимо написать args[0]. Знак + нужен для соединения строки "Hello World!" и первого параметра.

Если вы теперь нажмете F5, то увидите нечто страшное: вместо вывода сообщения, как в предыдущем примере, приложение MonoDevelop выдаст что-то вроде 'Unhandled exception: System.IndexOutOfRangeException' [Необработанное исключение: Выход за границы массива]. Так получилось потому, что args[0] ссылается на первый передаваемый в программу параметр, а в действительности мы его не передаем – ведь *MonoDevelop* просто запустил программу без параметров.

Решение состоит в том, чтобы открыть терминал и задать программе параметры вручную. В меню Applications (Приложения), выберите Accessories (Дополнительные), затем Terminal (Терминал). Теперь измените каталог на тот, в котором находится ваш проект. Внутри каталога вы увидите каталог **bin**, а также каталог **Debug**, куда *MonoDevelop* помещает исполняемые файлы, когда мы работаем над проектом. Внутри этого каталога будет находится ваша программа: ParamPrint.exe. Чтобы запустить ее, наберите команду:

```
mono ParamPrint.exe
```

Вы увидите то же сообщение об ошибке на консоли; а теперь попробуйте добавить параметр:

```
mono ParamPrint.exe LXF
```

На экран выведется 'Hello World: LXF'. А если вы попробуете

```
mono ParamPrint.exe Linux Format
```

на экран выведется только 'Hello World: Linux'. Строка не выведется целиком, так как каждый параметр считается отдельной строкой в массиве args, а параметры разделяются пробелом. В приведенном выше примере **Linux** – это args[0], а **Format** – это args[1]. Если хотите вывести всю строку, то наберите команду вот так:

```
mono ParamPrint.exe "Linux Format"
```

Кавычки сообщают *Bash* (это программа-оболочка), что **Linux Format** – это один параметр.

Пока все хорошо

На этом уроке было дано введение в MonoDevelop, вы узнали о пространствах имен и методах, какова структура кода C# и как выводить сообщения на экран. Также мы затронули объектно-ориентированное программирование, довольно сложный для изучения вопрос. Но главное – вы написали первую программу. Она не делает ничего полезного, но научила вас, как приступить к программированию. В следующий раз мы рассмотрим задачи посерьезнее, тогда-то и начнется веселье. LXF

Скорая помощь



Создав законченную программу, которую можно и распространять, щелкните на меню с надписью Debug, и измените значение на Release. Тогда при компиляции программы она будет оптимизирована по скорости работы и размеру исполняемого файла.

Скорая помощь

Как пункт Build (F8), так и Build And Run (F5) доступны из меню Project.

» Через месяц Файловые системы: займемся файлами и каталогами. До встречи!

Настоящая безопасность Как воспользоваться умными утилитами Linux и защитить вашу машину.

Безопасность: Строим экран

ЧАСТЬ 4 Любой подключенный к сети компьютер открыт для атаки. **Д-р Крис Браун** научит, как уменьшить вашу уязвимость, создав заслон при помощи стандартных утилит Linux.



В речи по поводу Дюнкеркской эвакуации, перед тем, как сказать «Мы будем сражаться на пляжах», Уинстон Черчилль изрек «Мы защитим наш остров любой ценой». Компьютеры – наши прекрасные маленькие островки, но они всегда находятся под обстрелом, о чем свидетельствуют журналы любого Linux-сервера, имеющего внешний IP-адрес.

Межсетевые экраны (брандмауэры) – один из наиболее важных рубежей обороны вашего компьютера. Это сетевые устройства (как аппаратные, так и программные, запускаемые на обычном компьютере), контролирующие трафик между внутренней сетью и Интернетом. Есть несколько технологий создания таких экранов. Например, запросы от клиента по протоколам прикладного уровня (HTTP, FTP или DNS) перенаправляются через прокси-сервер; таким образом, прямое соединение между узлом локальной сети и Интернетом становится ненужным. Трансляция сетевых адресов (NAT или маскарадинг) также делает вашу сеть невидимой снаружи, позволяя еще и использовать свой диапазон адресов внутри сети. Однако наш сегодняшний урок посвящен классическому пакетному фильтру, работающему на сетевом уровне (IP). Я покажу вам, как настроить его в Linux.



Наш эксперт

Д-р Крис Браун – независимый инструктор по Linux, имеет степень доктора наук по физике элементарных частиц,

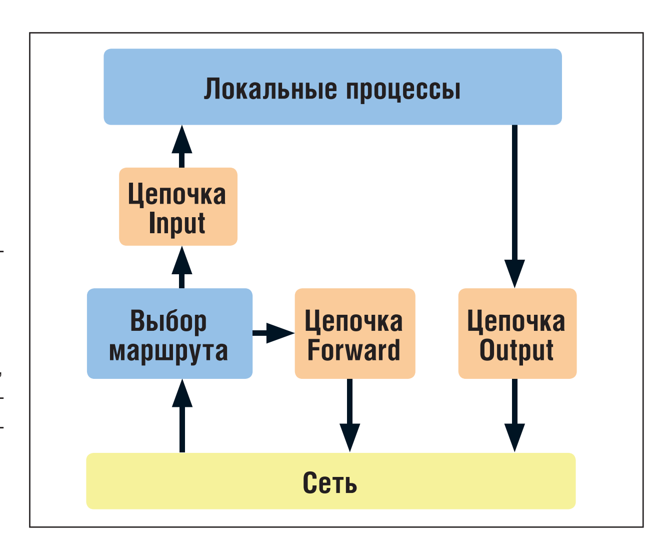
Часть 1: Разбираем iptables

В недрах ядра скрывается кусок кода под названием *netfilter* (сетевой фильтр) – он выполняет фильтрацию IP-пакетов: проверяет каждый исходящий и входящий пакет и решает его судьбу, основываясь на его параметрах, таких, как исходные и конечные адреса и порты или флаги заголовка TCP.

Два главных действия, выполняемых *netfilter* – принять пакет или, соответственно, отклонить его. Он также может вести журнал с помощью сервиса *syslogd*. Используя *netfilter*, можно настроить Linux-машину как межсетевой экран с фильтрацией пакетов.

Обычно в роли межсетевого экрана выступает машина, работающая между внутренней сетью предприятия и большим и злым Интернетом и защищающая данные корпоративных компьютеров от взломщиков, коварных скриптописцев и вирусов. В данном случае фильтрация применяется к пакетам, приходящим извне и переправляемым во внутреннюю сеть. Можно также фильтровать пакеты, идущие на межсетевой экран изнутри или исходящие из него; то есть использовать *netfilter* как персональный брандмауэр, пригодный даже для одиночной домашней машины, подключенной к сети через ADSL или модем.

Netfilter настраивается с помощью специальных правил, задаваемых из консоли командой *iptables*. Ее синтаксис – полноценный язык



› **Рис. 1. Три стандартных цепочки позволяют фильтровать входящие, исходящие и проходящие пакеты.**

» **Месяц назад** Мы использовали Nmap, Sussen и Nessus для жесткого поиска уязвимостей.



межсетевой на базе Linux

Зачем вам брандмауэр?

Интернет, увы, становится враждебным окружением, к которому уже нельзя подключаться непосредственно, без брандмауэра (он же firewall). Однако домашние пользователи с широкополосным доступом должны различать две вещи. Если ADSL-модем воткнут в вашу машину напрямую (ну, или через USB), то у нее есть реальный IP-адрес, к которому можно подключиться извне, и она нуждается в защите.

А если вы подключаетесь к внешнему ADSL-модему через Ethernet, то в такой модем уже встроен маршрутизатор, выполняющий преобразование адресов (Network address translation – NAT). Большое преимущество NAT состоит в том, что машинам из внешнего мира запрещено подключение к внутренним – т.е. к вашей домашней машине [по такому принципу работают многие российские интернет-провайдеры, однако, имейте в виду, что в этом случае ваши соседи по сегменту также потенциально опасны – кое-кто из них, например, не побрезгует воспользоваться «дырой» в чужой системе, чтобы посидеть в Интернете за ваш счет, – прим. ред.].

со своими правилами. Типичное правило включает в себя компоненты для распознавания определенных пакетов и заканчивается директивой принятия или отклонения подобного пакета. Вот пример того, как это выглядит:

```
iptables -A INPUT -i eth0 -p udp -s $NAMESERVER --sport 53 -d
140.116.5.1 --dport 53 -J ACCEPT
```

Чтобы разобраться в этом, нужно представить, какими путями пакеты приходят, проходят и уходят с вашей машины. Есть три варианта, показанные на **рис. 1**. Входящий пакет проверяется пакетным фильтром, и следует его решению. Если адрес доставки пакета соответствует данной машине, пакет проходит через входную цепочку (Input) и идет вверх по стеку протоколов. Если пакет предназначен для другой машины, реализация протокола сетевого уровня опрашивает таблицу маршрутизации, чтобы узнать, на какой сетевой интерфейс его отправить. Затем он идет по цепочке Forward и возвращается в сеть. Наконец, пакеты, формируемые на данной машине, проходят через выходную цепочку (Output) и попадают в сеть. Каждая из этих цепочек, в сущности, является простым набором условий, через проверку которых и проходит пакет.

Настроим индивидуальные фильтры

Теперь вы понимаете, что в примере выше мы добавляем правило во входную цепочку (-A INPUT). Условие гласит, что пакет должен придти на сетевой интерфейс eth0, и это должен быть UDP-пакет. Оно также указывает исходные адрес и порт, а также адрес и порт назначения, которым должен соответствовать пакет. Эта информация передается в заголовке пакета. Завершающая часть (-J ACCEPT) говорит *netfilter*, что делать с тем пакетом, который соответствует всем указанным в правиле параметрам. Эта часть называется целью условия. Возможные цели таковы:

- ACCEPT – принять пакет;
- DROP – молча отклонить пакет;
- REJECT – отклонить пакет и сказать об этом отправителю;
- LOG – записать прибытие пакета (и позволить ему пройти к следующему условию).

Списки условий для цепочки могут быть весьма длинными: 50 – обычнейшее количество. Для каждого пакета проверяется каждое условие. Как только произойдет совпадение, будет предпринято соответствующее решение (цель), и следующие условия проверяться уже не будут. Если пакет доходит до конца цепочки, не удовлетворив ни одному условию, его судьба зависит от «политики» цепочки. Например,

```
iptables -P INPUT DROP
```

говорит, что все такие пакеты будут отклонены. Простейший способ создания набора условий – установить политику по умолчанию в DROP, а затем добавлять условия для нужных пакетов. Это подход с «презумпцией виновности». Другой подход (установка политики по умолчанию в ACCEPT, а затем создание правил для блокировки ненужных пакетов) гораздо сложнее, менее безопасен и не рекомендуется.

Вы можете улучшить организацию ваших правил, определив собственные цепочки и присвоив им имена. Например, я могу определить цепочки TCP_RULES и UDP_RULES. Затем, в главной входной цепочке я могу задать всего два предопределенных правила:

```
iptables -A INPUT -p tcp -J TCP_RULES
iptables -A INPUT -p udp -J UDP_RULES
```

Это позволяет более гибко управлять наборами правил, да и более эффективно; например, UDP-пакет никогда не будет сравниваться с правилами из цепочки TCP_RULES. Я предпочитаю думать, что прыжки по цепочке аналогичны проходам по процедурам (подпрограммам).

Механизм «попакетной» проверки IP-трафика – лишь половина умений *iptables*. Возможно еще записывать, когда происходит TCP или UDP-транзакция, и проверять пакеты не только по их заголовкам, но и в контексте совершаемого соединения... Борюсь с искушением вдаться в подробности, иначе урок разросся бы до размеров журнала.

Можно настроить межсетевой экран путем создания полного набора правил и команды *iptables*, как было показано. Начните с политики безопасности, определяющей, какие сервисы должны быть доступны, вычислите, какой трафик они генерируют, поместите соответствующие правила в цепочки, а все остальное запретите. Все это требует отличного знания TCP/IP и большой внимательности. Построение межсетевого экрана таким способом аналогично созданию большого web-сайта путем ручной верстки в *vi* или написанию программ на Ассемблере. Другими словами, это лучше оставить экспертам, которым нужен полный контроль над каждой настройкой.

Но если вы все-таки решите дерзнуть, ознакомьтесь с материалом по *iptables* из LXF47. Мы положили его на диск в формате PDF. »

Скорая помощь

Будьте осторожны, когда настраиваете брандмауэр на машине, к которой у вас нет физического доступа. Очень легко заблокировать все входящие соединения, задавая политику по умолчанию: вы и охнуть не успеете. Верьте мне! Я уже пробовал…

Рекомендуется прочесть

Лучшая книга по брандмауэрам в Linux, конечно же, *Linux Firewalls* Стива Сьюринга [Steve Suring] и Роберта Циглера [Robert Ziegler], третье издание (и, будете смеяться, третий издатель, Novell Press). Эта книга не только подробнейше описывает использование *iptables*, но и рассматривает внутреннюю защиту, SELinux и мониторинг сети.

Часть 2: Простой путь

Я показал вам продвинутый путь настройки межсетевого экрана, однако большинство из нас (включая меня), возможно, предпочтут использование утилиты, которая позволит указать политику безопасности на более высоком уровне и сгенерирует команды *iptables* сама. А потом их можно будет подредактировать, как описано в первой части.

Мне нравится модуль конфигурации брандмауэра *YaST* в SUSE. Он позволит вам настроить брандмауэр примерно на том же уровне, на котором вы задаете политику безопасности, в противоположность уже рассмотренному нудному процессу ручного ввода команд. Модуль требует указать для каждого сетевого интерфейса три зоны: внешнюю, внутреннюю или демилитаризованную. Эти термины иллюстрируются **рис. 2**, где показана архитектура классического межсетевого экрана в корпоративной сети.

Внешние интерфейсы – те, что подключены к огромному злому Интернету; в домашних условиях это обычно ADSL или простой модем. Если у вас всего одна машина, ваш сетевой интерфейс будет внешним.

Внутренние сетевые интерфейсы – те, что подключены к доверенным узлам. В небольшом офисе, где как шлюз используется Linux-машина, это будет интерфейс, подключенный к локальной сети.

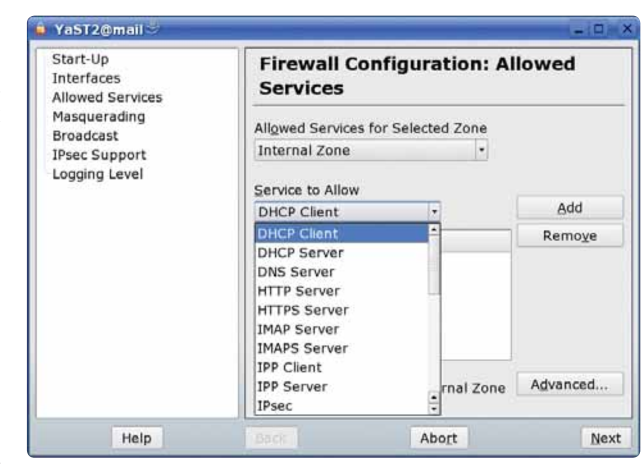
Демилитаризованная зона (DMZ) – это сеть, в которой находятся видимые снаружи машины, например, Web/FTP/почтовые-серверы. Сказать по правде, если вы настраиваете межсетевой экран для использования в корпоративной среде с DMZ, вы обязаны изучить более глубокие материалы по данной тематике. Тот же модуль *YaST*, к примеру, недостаточно гибок для настройки направления пакетов между внутренней сетью и DMZ; он всего лишь определяет правила ограничения доступа к машине из всех трех зон.

Если интерфейс у вас на компьютере только один, не имеет значения, считаете ли вы его внешней или внутренней зоной. Назначьте зону произвольно и определите доступные сервисы, как мы покажем далее.

Определив, какие сетевые интерфейсы соответствуют нужным зонам, переходите на экран доступных сервисов для каждой из зон (**рис. 3**). Слева вы видите перечень семи экранов модуля *YaST*, справа – выпадающие списки зон и сервисов (DNS, IMAP, HTTP и т. д.). Выберите зону и определите доступные ей сервисы.

Скорая помощь

Рискуя быть навязчивым, я хочу подчеркнуть важность политики безопасности. Пока вы не сядете и не зададитесь вопросом «Кто и что может делать с моей машиной?», вы не готовы настраивать правила межсетевого экрана, отключать ненужные сервисы и повышать безопасность компьютера.



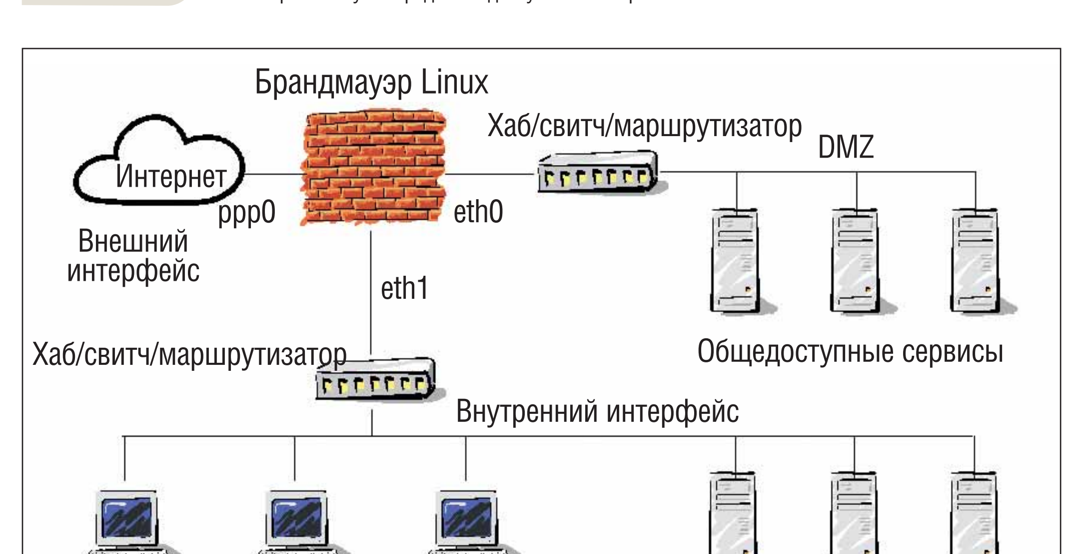
› **Рис. 3 Модуль YaST для настройки брандмауэра. Здесь указывается, какие сервисы доступны в каждой из трех зон.**

Для разрешения доступа к сервису, выберите его из списка и нажмите Add (Добавить). Чтобы запретить сервис, выделите его и нажмите Remove (Удалить). Для сервисов, которых нет в списке, вы должны будете указать номер порта. Например, для разрешения Telnet-сеансов (возможно, вы захотите разрешить их только для внутренней сети, поскольку Telnet небезопасен) нажмите на Advanced... и введите номер порта (23) в поле TCP Ports. Внутренняя зона должна обрабатываться по-особому. Внизу вы найдете флажок Protect Firewall From Internal Zone. Пока он не отмечен, к пакетам, исходящим из внутренней сети, не будут применяться никакие правила.

За кулисами

Модуль *YaST* не генерирует правила *iptables* напрямую. Вместо этого он редактирует файл **/etc/sysconfig/SuSEfirewall2**. Если у вас SUSE, рекомендую изучить этот файл. Он очень хорошо прокомментирован и углубит ваше понимание действий *YaST*, а также предоставит синтаксически более сложные примеры.

Сам межсетевой экран настраивается на раннем этапе загрузки через два скрипта, *SuSEfirewall2_init* и *SuSEfirewall2_setup*, находящиеся в директории **/etc/init.d**. Первый запирает брандмауэр (пропуская только трафик bootp и ping), а второй, запускающийся несколько позже, устанавливает цепочки правил при включении брандмауэра и очищает их при отключении. Оба скрипта в конечном итоге вызывают **/sbin/SuSEfirewall2**: это движок механизма межсетевого экрана в SUSE, и в нем генерируются команды *iptables*. Я бы не рекомендовал вам в нем копаться (особенно после плотного обеда), если вы не любитель скриптоужастиков.



› **Рис. 2. Межсетевой экран с тремя интерфейсами защищает DMZ и внутреннюю сеть от внешнего мира. Для каждой сети можно выбрать свой уровень фильтрации.**

Брандмауэр Fedora

Не только в SUSE есть графические утилиты для настройки брандмауэра. На **рис. 4** показана утилита *system-config-securitylevel*, входящая в Fedora. Это более простой инструмент, чем модуль *YaST*. Она не позволяет определять зоны, а просто закрывает и открывает нужные порты, и подходит только для настройки личного брандмауэра на одиночной машине.

Предотвращение скрытого сканирования

В прошлом месяце я рассказывал о скрытом сканировании с помощью Nmap. Этот вид сканирования использует нештатные комбинации флагов TCP-пакетов, а скрытым называется, поскольку маловероятно, что системный журнал его зафиксирует. Однако, используя способность *netfilter* проверять флаги в заголовке TCP-пакета и записывать события в журнал, можно не только блокировать подобные попытки, но и регистрировать факт их наличия. Подробности довольно запутанны, но пример пары правил против FIN-сканирования прояснит суть этой идеи:

```
iptables -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix
“Stealth scan”
iptables -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j DROP
```

Первое правило служит для обязательной записи события в журнал. После цели LOG пакет продолжает движение по цепочке условий (в отличие от целей DROP и ACCEPT. Принятые или отклоненные пакеты на дальнейшую проверку не пойдут). В данном случае пакет, удовлетворяющий первому условию, удовлетворит и второму, согласно которому он будет отклонен. Параметры --tcp-flags ACK,FIN FIN описывают комбинацию TCP-флагов. Первый список состояний (ACK,FIN) перечисляет тестируемые флаги, второй (FIN) – те из них, что установлены. Таким образом, условие соответствует тем пакетам, в которых есть FIN-флаг, но нет ACK. При нормальном TCP-соединении эта комбинация невозможна, зато типична для скрытого сканирования.

Проведите эксперимент: если у вас две Linux-системы, выберите одну из них мишенью, а на второй запустите нечто вроде

```
nmap -sF -p1-50 192.168.0.3
```

(подставьте нужный IP-адрес). *Nmap* сообщит вам об открытых портах. Если вы проследите судьбу пакетов через *Ethereal*, то увидите, что FIN-пакеты достигли цели, а в ответ были отправлены пакеты RST,ACK. Теперь добавьте на системе-мишени два правила, показанных выше, и повторите попытку. Вы увидите, что *Nmap* больше не обнаруживает открытые порты, а в журнале (у меня это **/var/log/firewall**) появились новые сообщения. *Ethereal* покажет, что FIN-пакеты по-прежнему доходят, но не получают ответа. На подобных экспериментах можно научиться многому. Вот вам развлечение для дождливого вечера!

Скорая помощь

Сканер портов, типа *Nmap*, рассмотренного на прошлом уроке — отличная утилита для проверки корректной работы правил вашего межсетевого экрана.

Вы можете просмотреть ваши правила, выполнив команду *iptables-save* (ее вывод показан на **рис. 5**). Здесь вы видите пример определенной пользователем цепочки (RH-Firewall-1-INPUT), устанавливающей правила как для входящей, так и для исходящей цепочек. Если сохранить вывод *iptables-save* в файле, из него можно будет восстановить правила командой *iptables-restore*.

Netfilter имеет и другие возможности (например, NAT и фильтрацию по состояниям), я о них успел только намекнуть. Есть и другие методы, например, прокси-серверы уровня приложений, они тоже полезны при настройке межсетевого экрана. LXF

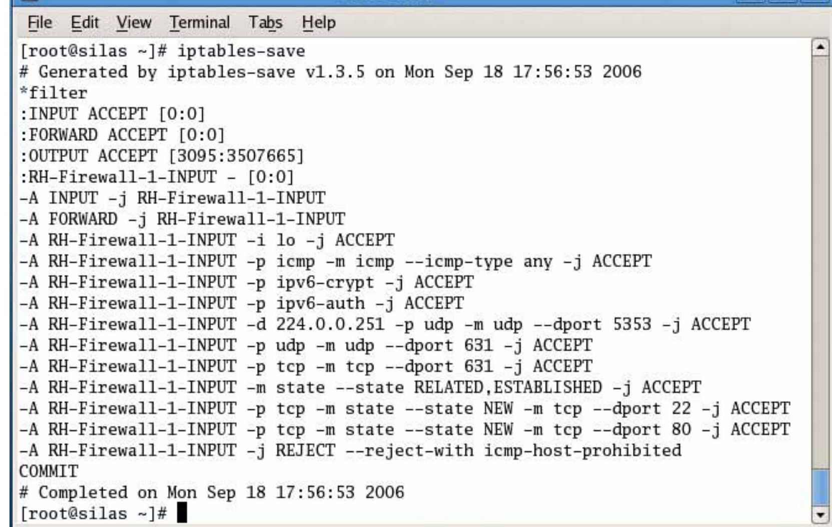


Скорая помощь

Если вы столкнетесь с проблемами, заставляя какой-либо сетевой сервис работать, стоит проверить, не стоит ли на его пути *netfilter*. Мне случалось потерять много времени, прежде чем я обнаруживал, что все дело в брандмауэре. Его отключение (ненадолго!) значительно упростит настройку.



› Рис. 4 Утилита настройки брандмауэра в Fedora сгодится при создании личного брандмауэра, но не более того.



› Рис. 5 Просмотр действующих правил командой *iptables-save*. Если сохранить их в файле, можно будет их восстановить командой iptables-restore.

» **Через месяц** Утилиты-детективы сообщат о подозрительных изменениях файловой системы.

DocBook: Пишем

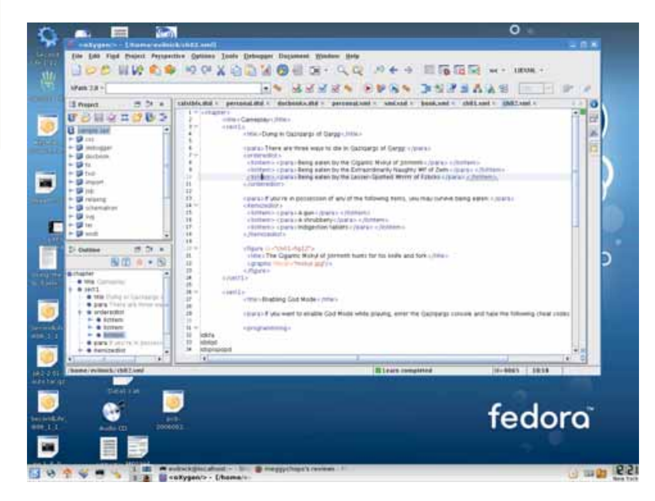
Что общего у ядра, FreeBSD, KDE и Gnome? Документация! **Пол Хадсон** рассказывает о новой технологии для ее написания.



Наш эксперт

Пол Хадсон не уважал XML, пока не встретил XPath, XInclude и XPointer. Теперь он использует XML даже чаще, чем родной английский.

Создание документации к программам всегда было проблемой для программистов: ведь это же тупость – объяснять, как все работает, когда оно ослепительно-очевидно. Наши программы, естественно, всегда понятны нам, потому что мы сами их написали; но есть люди, до сих пор думающие, что компьютеры и программы – это какое-то колдовство. Поэтому необходимо кратко объяснить им, какие кнопки нажимать для выполнения нужных операций. На этом уроке мы изучим DocBook – формат написания документации. Он основан на XML, и с ним можно работать в любом текстовом редакторе. Он используется во многих крупных проектах, включая ядро Linux, FreeBSD и KDE, поэтому рано или поздно вы с ним столкнетесь.



› **Использование хорошего XML-редактора типа *Oxygen* сэкономит массу времени. Подробности см. во врезке XML-редакторы на стр. 66.**

В процессе урока мы будем документировать выдуманную игру Ловля мух. Если у вас еще нет открытого проекта, который вы могли бы задокументировать, то присоединяйтесь к уже существующему, или начните новый!

Ныряем в DocBook

DocBook (как XML) имеет строгую структуру, которой необходимо следовать, но она довольно понятна. Помните, что DocBook – это разновидность XML, разработанная специально для создания документации, поэтому его структурными элементами являются главы, разделы, заголовки, абзацы и так далее. Вот и все, что необходимо знать для создания нашей первой части документации: книги, состоящей из одной главы. Начнем:

```
<?xml version="1.0" ?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML
V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<book>
  <title>Ловля мух!</title>
  <chapter id="ch01">
    <title>Введение</title>
    <sect1>
      <title> Добро пожаловать в лучшую в мире игру!</title>
      <para>Да, она действительно так хороша.</para>
    </sect1>
  </chapter>
</book>
```

[Необходимо использовать кодировку UTF-8 или указать ее явно в тэге <?xml?> при помощи атрибута 'encoding', – прим. ред.] Тип DTD, который мы собираемся использовать – стандарт DocBook 4.4, доступный из сотворившей его организации Oasis. Если хотите, можете скопировать файл **.dtd** на свою машину – это немного ускорит вашу работу, потому что в противном случае вашему компьютеру для валидации документа придется копировать DTD из сети.

Разберем написанное: наша полностью законченная документация называется книгой (book). Как и большинство книг, она состоит из отдельных глав, разбитых на разделы. В нашем примере с Ловлей мух у нас могут быть следующие главы: Введение, Описание игрового процесса, Многопользовательская игра, Разрешение проблем и Контактная информация.

Каждая глава состоит из разделов и подразделов, это упрощает чтение материала. Например, раздел Многопользовательская игра может состоять из секций 'Запускаем сервер', 'Подсоединяемся к серверу' и 'Настраиваем брандмауэр'. 'Запускаем сервер' можно затем подразделить на 'Выделенный сервер', 'Невыделенный сервер', 'Обнаружение вашего IP-адреса' и так далее. В терминологии DocBook эти разделы называются 'sections' и вы можете выбирать из <sect1> (раздел верхнего уровня), <sect2>, <sect3> или <sect4>. Обычно в оглавление попадают только первые три уровня – четвертый чуть больше размером, чем жирный текст, и если вы поймаете себя на том, что вовсю

» **Месяц назад** Мы изучали совместное использование файлов при посредстве Kamaelia.

документацию



используете <sect4>, то придется признать: ваша документация чересчур многословна!

Итак, книга <book> содержит главы <chapter>, которые содержат разделы <sect1>, содержащие подразделы <sect2>, в свою очередь, содержащие <sect3>, а те <sect4>. Вот как все может выглядеть:

```
<chapter id="ch01">
  <title>Введение</title>
  <sect1>
    <title> Добро пожаловать в лучшую в мире игру!</title>
    <para>Да, она действительно так хороша.</para>
    <sect2>
      <title>Почему же она так хороша?</title>
      <para>На это есть множество причин!</para>
      <sect3>
        <title>Взрослым...</title>
        <para>Много крови, убийств и жутких моментов!</para>
      </sect3>
      <sect3>
        <title>Детям...</title>
        <para>Еще больше крови, убийств и жутких моментов!</
para>
      </sect3>
    </sect2>
  </sect1>
</chapter>
```

Необходимо строго соблюдать иерархию структуры – вы не можете создать <chapter>, затем сразу <sect3>, или же <sect3>, а затем <sect2>, как показано ниже:

```
<chapter id="ch01">
  <sect3>
    <sect2>
      <title>Введение</title>
    </sect2>
  </sect3>
</chapter>
```

И книга, и глава, и раздел могут иметь свой собственный <title>, то есть заголовок. После этого начинается основная работа: множество элементов <para>, каждый из которых представляет один текстовый абзац.

Как вы понимаете, разобраться с XML здесь не самое сложное – куда сложнее написать качественную документацию!

Структура абзаца

Набор бесформенных абзацев – занятие нудное, но это легко исправляется тем, что DocBook предоставляет специальные тэги для форматирования. Кроме базовых элементов, типа нумерованных списков и выделения жирным шрифтом или курсивом, вы можете особо выделить листинги программ, цитаты других людей, экранные снимки и многое другое. У DocBook есть тэги для всего, что как-то связано с программами или компьютерами, так что он далеко не прост!

На наше счастье, достаточно знать только о тех элементах, которые будут вами использоваться. Основные элементы можно разделить на две категории: встраиваемые и блочные. Блочные элементы образуют естественные части текста, которые в результирующем документе отделяются пустой строкой. Встраиваемые элементы изменяют отдельные слова внутри блочных, не влияя на организацию текста. Начнем с основных элементов:

```
<para>Чтобы организовать собственный сервер Ловли, вы
  <emphasis>должны</emphasis> открыть <acronym>TCP</acronym>
  порт 556 в брандмауэре. <emphasis role="bold">Внимание:</
  emphasis> мы не отвечаем за возможные последствия.</para>
<para>Вы можете найти предустановленные настройки брандмауэра в
файле <filename>firewall.config</filename> в каталоге с игрой.</para>
<para>Если проблемы все еще есть, попытайтесь подключиться к
нашему сайту для проверки брандмауэра
  <systemitem role="url">http://www.qaziqargs.com/firewall</
  systemitem>.</para>
```

Тэг <emphasis> задает либо жирный шрифт, либо курсив: если вы уточните его словом 'bold', то текст будет выделен жирным; в противном случае – курсивом. Элемент <systemitem> – довольно хитрый зверь: в зависимости от роли, он может хранить IP-адреса, доменные имена, имена пользователей и, как в моем примере, URL-ы. Кому интересно, знайте, что нет никаких способов указать «текст» ссылки (то есть фразу, которой она будет представлена в документе), потому что DocBook спроектирован для работы с любыми носителями, включая печатные (где, понятное дело, щелкать по URL бессмысленно!).

Мы уже рассмотрели тэги <para> и <title>, однако интерес представляют еще пять тэгов: <orderlist>, <itemizedlist>, <listitem>, <programlisting> и <screen>. Первые три связаны между собой, поэтому начнем с них: покажем, как сделать упорядоченный (нумерованный) и неупорядоченный список (где порядок элементов не важен):

```
<para>Существует три способа погибнуть в игре Ловля мух:</para>
<orderedlist>
  <listitem><para>Быть съеденным лягушкой</para></listitem>
```

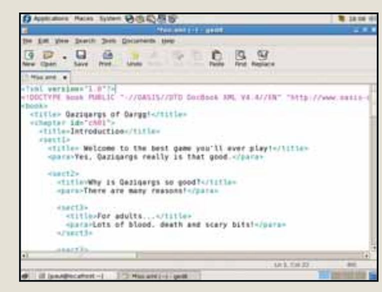
»

Скорая помощь

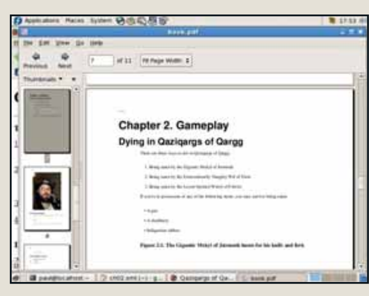
Вы можете использовать xmllint с параметром -o, чтобы сохранить вывод в файле XML. Это особенно полезно, когда используется параметр --xpointer, так что xmllint выполняет директивы XInclude, а затем сохраняет скомбинированный файл.

DocBook в роли нормативного формата

DocBook обычно считается «нормативным» форматом, то есть он не предлагается конечному потребителю. Для просмотра его необходимо преобразовать в другой формат. Это позволяет вам сосредоточиться на документации, а не на ее представлении.







› DocBook XML (слева) как нормативный формат означает, что его можно конвертировать как в HTML (посередине), так и в PDF (справа).

```
    <listitem><para>Быть съеденным птицей</para></listitem>
    <listitem><para>Быть прихлопнутым мухобойкой</para></listitem>
</orderedlist>
<para>Если у вас в наличии один из следующих предметов, вы можете избежать смерти:</para>
<itemizedlist>
    <listitem><para>Ружье</para></listitem>
    <listitem><para>Кустарник</para></listitem>
    <listitem><para>Слабительное</para></listitem>
</itemizedlist>
```

Будь это HTML, то <orderlist> был бы <ol>, <listitem> стал бы <li> и так далее – невелика разница, только что тэги DocBook длиннее! Текст внутри <listitem> должен обрамляться тэгами <para>.

Элементы <programminglisting> и <screen> используются подобным образом, но они являются отдельными сущностями и могут быть по-разному отформатированы при выводе, если потребуется. Например:

```
<para>Для включения режима бессмертия в игре, войдите в консоль Ловли и наберите следующие коды:</para>
<programlisting>
idkfa
iddqd
idspispopd
</programlisting>
<para>Вы узнаете, что режим бессмертия активирован, когда увидите следующее
сообщение на экране:</para>
<screen>
Сообщаем:
режим бессмертия активирован!
</screen>
```

Все символы-разделители сохраняются в элементах <programminglist> и <screen>, так что набранный вами текст будет отображен на экран в таком же виде.

Скорая помощь

Если вы хотите сравнить два XML-документа, используйте *xmldiff*, а не обычную утилиту *diff*. *xmldiff* запрограммирована так, чтобы находить разницу в структуре, а не просто разницу текстов.

Работа с несколькими главами

Прежде чем начать печатать, подумайте об организации работы. Прекрасно, что вы можете разбить вашу книгу на главы, разделы и абзацы, но важно разделить на части и работу, создав предпосылки для привлечения команды. То есть – использовать несколько файлов, разделенных согласно вашим нуждам – например, можно выделить по файлу для каждой документируемой функции.

Файл, в который включают другие файлы, использует стандарт XInclude, в котором вы можете определить URL, загружаемый при обработке файла. Это позволит вам переложить часть работы на коллег, а затем собрать все документы вместе. Чтобы распределить главы по отдельным файлам, ваша XML-книга должна выглядеть примерно так:

```
<?xml version="1.0"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<book>
    <title> Ловля мух!</title>
    <include xmlns="http://www.w3.org/2001/XInclude" href="chapter01.xml"/>
    <include xmlns="http://www.w3.org/2001/XInclude" href="chapter02.xml"/>
    <include xmlns="http://www.w3.org/2001/XInclude" href="chapter03.xml"/>
    <include xmlns="http://www.w3.org/2001/XInclude" href="chapter04.xml"/>
</book>
```

Начало положено, но это не облегчит людям чтение: если вы скажете «смотри раздел 3 главы 4», то им придется искать, где находится это место. Обычно DocBook-конверторы создают оглавление, содержащее номера страниц (или HTML-ссылки), но такой подход требует, чтобы пользователи поднимались к оглавлению, находили ссылку, переходили по ней, а затем нажимали кнопку Назад два раза, чтобы вернуться туда, откуда пришли.

Мы можем облегчить им задачу, используя тэг <xref>, позволяющий делать ссылки на главы, разделы и некоторые другие блоки. Взглянув на наш первый пример с книгой, вы заметите строку:

```
<chapter id="ch01">
```

Именно атрибут "id" позволяет нам устанавливать ссылки. Например, если мы хотим сделать ссылку на главу 1, то можем написать следующее:

```
<para>Если ваша игра не устанавливается, вернитесь и прочтите инструкции в <xref linkend="ch01" />.</para>
```

Нам не требуется писать «прочтите инструкции в главе 1» – такой текст будет получен после преобразования документа в требуемый формат. Например, если документ преобразуется в HTML, <xref> станет гиперссылкой с текстом типа «главе Руководство по установке», указывающей на начало этой самой главы.

Можно применить атрибуты id и к разделам, но когда вы делаете ссылки на отдельные блоки <para>, ссылки обычно создаются на начало раздела, содержащего требуемый блок.

Оформление страницы

Документация – это не только текст. На самом деле, из проповедей Кэти Сиерра вы узнаете, что текст – лишь небольшая часть вашей работы! DocBook позволяет добавлять таблицы и картинки, а если вы добавите к ним атрибут id, то сможете ссылаться на них с помощью тэга <xref>.

Чтобы вставить рисунок, нам необходим сам рисунок и подпись к нему. Вот как это выглядит в DocBook XML:

```
<figure id="ch01-fig12">
    <title>Муха-гигант охотится за человеком</title>
    <graphic fileref="figs/mxkyl.png"/>
</figure>
```

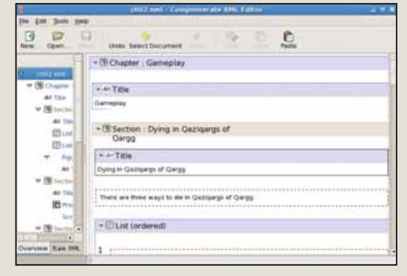
Обратите внимание на атрибут id: сначала указан номер главы, а затем номер рисунка. Такой формат необязателен, но с ним удобнее отслеживать рисунки в большом проекте.

Работать с таблицами немного сложнее, потому что нам необходимо определить тип требуемой таблицы, указать, сколько в ней столбцов и строк и, наконец, ввести содержимое ячеек. Простейшая таблица выглядит так:

```
<informaltable>
    <tgroup cols="2">
        <tbody>
        <row>
            <entry>F1</entry>
            <entry>Помощь</entry>
        </row>
        <row>
            <entry>F2</entry>
            <entry>Сменить оружие</entry>
        </row>
        </tbody>
    </tgroup>
```

XML-редакторы

Вы можете набирать XML, используя любой текстовый редактор – это одно из его преимуществ. Некоторые текстовые редакторы (вроде *Kate*) умеют подсвечивать синтаксис, что позволяет выявить ошибки. Другие, типа Conglomerate, просто путаются под ногами со своими ошибками. Редактор *Oxygen XML* получил 9/10 в нашем обзоре на стр. 14 – посмотрим, что скажете вы!



› **Conglomerate: зачем писать документ за час? Лучше потратить три.**

```
</informaltable>
```

В терминах HTML, <informaltable> – это <table>, <row> – <tr>, а <entry> – <td>. Мы можем превратить нашу <informaltable> в обычную таблицу <table> с добавлением заголовка:

```
<table>
  <title>Клавиатурные сокращения</title>
  <tgroup cols="2">
```

Исходя из наличия элемента <tbody>, можно предположить, что должен быть и элемент <thead>, определяющий заголовки столбцов. Наша таблица хранит названия клавиш и их действия; ее можно модифицировать следующим образом:

```
<table>
  <title>Клавиатурные сокращения</title>
  <tgroup cols="2">
    <thead>
      <row>
        <entry>Клавиша</entry>
        <entry>Действие</entry>
      </row>
    </thead>
    <tbody>
```

Как <thead>, так и <tbody> существуют и в HTML (впрочем, используются там нечасто), так что они могут быть вам уже знакомы.

Проверка и преобразование

На вашей улице праздник: документация готова! Все, что осталось сделать – это раздобыть пользователей, способных в уме преобразовать XML в человеко-читаемый вид. Или поступить умнее: выполнить это преобразование автоматически. Если вы не издатель, то, скорее всего, выберете в качестве выходного формата HTML, тем более, что средства, позволяющие выполнить эту работу, наверняка у вас уже есть.

В Linux это инструменты *xmllint* (часть *libxml2*library, включенной на наш DVD) и *xmlto*. Первая утилита выполняет проверку синтаксиса и валидацию вашей книги. Вторая утилита – XML-конвертор, она позволит создать HTML из DocBook. Обе утилиты доступны в большинстве дистрибутивов (включая Fedora, SUSE и Ubuntu), хотя вы можете найти, что *xmlto* требует довольно много места на диске, поскольку устанавливает LaTeX (*см. стр. 86*), необходимый для создания выходных файлов в формате PDF.

Для начала проверим наш XML-файл:

```
xmllint --noout ch01.xml
```

Параметр --noout предписывает не выводить ваш XML-файл на экран, а печатать только возможные ошибки. Если XML-файл корректен, то на экран ничего не выведется. С помощью этой команды *xmllint* проверяет, является ли **ch01.xml** правильно сформированным XML [все тэги закрыты и т.п., – прим. ред.]. Однако он не проверит, является ли этот документ правильным документом DocBook XML [<sect3> всегда идет после <sect2> и т.п., – прим. ред.]. Чтобы это сделать, необходимо запустить команду:

```
xmllint --valid --noout book.xml
```

Теперь *xmllint* скопирует из сети DTD и выполнит валидацию на его основе. На этот раз вы можете увидеть кучу ошибок. Что же случилось? Наш XML-файл – это контейнер для четырех глав, использующий XInclude, который является отдельным стандартом. Пытаясь проверить Xinclude на соответствие DocBook DTD, мы не получим ничего, кроме ошибок.

Решение – попросить *xmllint* обработать директивы XInclude (то есть включить XML-файлы в **book.xml**), а уж затем проверять получившийся документ. Вот команда, которая это осуществляет:

```
xmllint --xinclude --postvalid --noout book.xml
```

Если все будет хорошо, никаких сообщений от *xmllint* не появится – это значит, что ваш XML-документ прошел валидацию и готов к выводу!

Использовать конвертор *xmlto* легко: укажите желаемый тип файла для вывода, а потом имя файла. Об Xinclude не волнуйтесь, все будет сделано автоматически. Для преобразования документа в формат, например, PDF, команда будет такой:

```
xmlto pdf book.xml
```

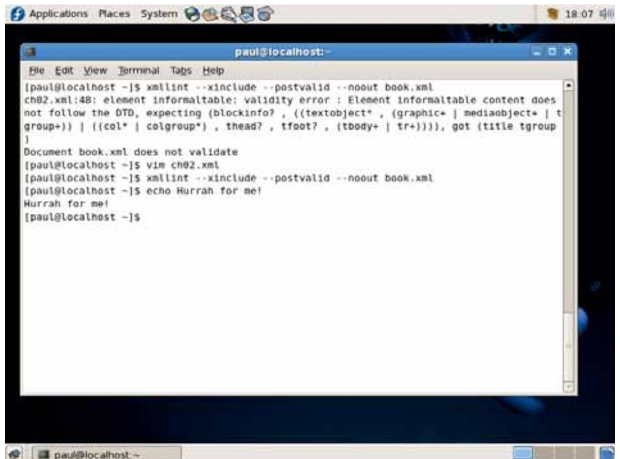
Вместо pdf можете подставить одно из следующих значений: html (каждая глава будет находиться в собственном файле), html-nochunks (весь текст будет помещен в один документ), man, txt (обычный текст), xhtml и xhtml-nochunks.

Ложка дегтя в *xmlto* – отсутствие «красивого» вывода (pretty print). Проблема невелика, так как в web-браузере HTML-формат смотрится нормально, но если вы собираетесь вручную редактировать HTML-код, она может стать большой.

Для ее решения существует программа *tidy*, имеющаяся в менеджере пакетов большинства дистрибутивов. Небольшая программа призвана получить на вход набитый тэгами (X)HTML файл и реализовать все эти табуляции, переводы строки и верхние регистры. Установив tidy, запустите ее следующим образом:

```
tidy -mqci book.html
```

Параметр m сообщает *tidy*, что файл **book.xml** надо модифицировать прямо на месте, q подавляет вывод ненужных сообщений, c очищает код, а i вставляет отступы, создавая визуальную структуру. Команда отработает, и ваш труд окончен: документация написана, преобразована в HTML и подготовлена для дальнейшего выпуска. Можете развалиться в кресле, ожидая аплодисментов… LXF



› **Если в документе есть ошибки, xmllint вас обругает.**

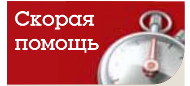
Печатаем код

Если ваш код или экранный вывод включает символы, которые поставят XML в тупик (а именно <, > или "), то лучше обрамлять их тэгом CDATA – это XML-тэг для необрабатываемых символьных данных. Вот как это выглядит:

```
<programlisting>
<![CDATA[
set Name = "<b>Квадзилла</b>";
]]>
</programlisting>
```

То, что внутри CDATA, не игнорируется (в смысле, идет на вывод), но и не обрабатывается как XML.

Скорая помощь

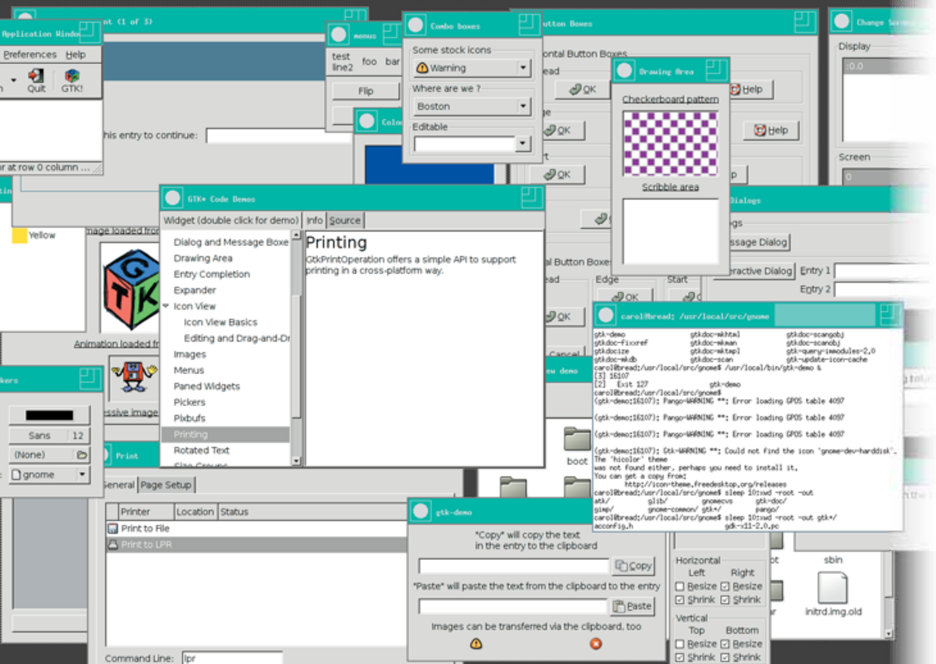


Вы можете заставить tidy автоматически преобразовывать HTML в XHTML, если хотите, но для этого ей может понадобиться CSS.

» **Через месяц** Мы научимся собирать свое собственное ядро.

Интернационализация

ЧАСТЬ 2: Сегодня **Андрей Боровский** научит приложения GTK+ говорить по-русски и нажимать на несколько кнопок одновременно



Проникновенье наше по планете особенно заметно вдалеке...

В. Высоцкий

В любом увлекательном деле, даже в таком, как программирование, есть своя рутина. Для меня такой рутиной всегда была интернационализация. Был бы я американским культурным империалистом, я бы вообще не обращал на нее внимания. Но я не американец и не империалист (даже в смысле культуры), а потому первая половина этой статьи будет посвящена тому, как научить программы *GTK+* разговаривать на разных языках, иначе говоря, интернационализации приложений.

Для интернационализации приложений *GTK+* мы воспользуемся пакетом *GNU gettext*, так что тем из вас, кто хорошо знаком с ним, будет достаточно беглого взгляда на приведенный ниже листинг программы, чтобы понять, что мы делаем. Для тех, кто не знаком с пакетом *gettext*, будут даны краткие, и никоим образом не исчерпывающие, пояснения. Более глубокое понимание работы утилит интернационализации GNU вы получите, ознакомившись со специальной документацией, которую можно найти, например, по адресу **www.gnu.org/software/gettext/manual/**.

Для тех, кто совсем не знаком с основами процесса перевода приложений Linux на разные языки, я кратко изложу базовые принципы. В процессе разработки программы весь текст интерфейса программы (названия кнопок и пунктов меню, текст диалоговых окон, сообщения об ошибках) пишется на одном языке, как правило, на английском. При этом дальновидный программист заранее готовит интерфейс программы к переводу на другие языки (процесс подготовки программы к «многоязычию» и называется интернационализацией). Программист добавляет в программу специальный код, отвечающий за работу с разными языками, а также помечает все строки интерфейса программы, которым потребуется перевод, специальными маркерами (макросами). Строка, помеченная для перевода, может выглядеть например так:

```
g_print(gettext("Translate this!"));
```

Здесь gettext() – это макрос, который указывает, что необходимо перевести строку *Translate this!*. Затем программист сканирует исходные тексты программы с помощью утилиты *xgettext*. Утилита *xgettext* копирует все строки, помеченные для перевода, в специальный файл. Далее начинается процесс локализации, то есть адаптации программы к конкретной локали. С помощью специальной утилиты, например *KBabel*, на основе файла, полученного от *xgettext*, создаются файлы перевода интерфейса, в которых каждой оригинальной строке сопоставлен перевод на другой язык (такие файлы называют каталогами сообщений). Для того, чтобы интернационализированная программа могла воспользоваться каталогами сообщений, их нужно скомпилировать в специальный «машинный» формат с помощью утилиты *msgfmt*. Полученные в результате двоичные каталоги сообщений распространяются вместе с двоичным файлом приложения. Во время выполнения программа определяет текущую локаль и загружает двоичный файл, содержащий перевод сообщений интерфейса на соответствующий язык. В ходе работы программы все строки интерфейса, для которых подготовлен перевод, заменяются локализованными вариантами. Каким образом выполняется замена строк? За нее отвечают те самые макросы, которые по совместительству служат маркерами. В приведенном выше примере во время выполнения программы макрос gettext() попытается найти перевод строки *Translate this!*. Если перевод будет найден, аргументом функции g_print() станет строка перевода, если же перевод для данной строки найден не будет, макрос передаст функции g_print() исходную английскую строку.

Возможно, эта схема покажется вам слишком сложной, но я могу вас успокоить. Во-первых, на практике все выглядит проще, чем в описании (сейчас мы перейдем к примеру, и вы сами это увидите). Во вторых, распространение файлов, содержащих перевод интерфейса отдельно от исполнимых файлов программы, представляет собой очень мощный механизм, благодаря которому у локализаторов программы появляется возможность переводить ее на другие языки, не прикасаясь к исходным текстам. Возможность подключить множество людей к процессу локализации вашей программы стоит того, чтобы выполнить пару лишних утилит.

Месяц назад Мы создали наше первое приложение GTK+.

И компоновка



Теперь я призываю расслабиться тех, кто устал от теории и вновь подключиться тех, кто ее пропустил, потому что мы переходим к практическому примеру интернационализации приложения *GTK+*. В качестве подспорья для интернационализации мы воспользуемся программой *helloworld* из LXF86. Начнем мы с того, что внесем некоторые изменения в исходный текст программы (вы найдете его на диске в файле **helloworld.c**):

```
#include <gtk/gtk.h>
#include <libintl.h>
#define _(String) gettext (String)
#define gettext_noop(String) String
#define N_(String) gettext_noop (String)
#define GETTEXT_PACKAGE "helloworld"
#define LOCALEDIR "./locale"
static void button_clicked(GtkWidget * widget, gpointer data)
{
  g_print("Button pressed!\n");
}
static gboolean delete_event(GtkWidget * widget, GdkEvent * event,
gpointer data)
{
  g_print("Delete event occurred\n");
  return FALSE;
}
static void destroy(GtkWidget * widget, gpointer data)
{
  g_print("Destroy signal sent\n");
  gtk_main_quit();
}
int main(int argc, char ** argv)
{
  GtkWidget * window;
  GtkWidget * button;
  bindtextdomain (GETTEXT_PACKAGE, LOCALEDIR);
  bind_textdomain_codeset (GETTEXT_PACKAGE, "UTF-8");
  textdomain (GETTEXT_PACKAGE);
  gtk_init(&argc, &argv);
  window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
  gtk_window_set_title(GTK_WINDOW(window), _("Hello World!"));
  gtk_container_set_border_width(GTK_CONTAINER(window), 10);
  g_signal_connect(G_OBJECT(window), "delete_event", G_
CALLBACK(delete_event), NULL);
  g_signal_connect(G_OBJECT(window), "destroy", G_
CALLBACK(destroy), NULL);
  button = gtk_button_new_with_label(_("Quit"));
  g_signal_connect(G_OBJECT(button), "clicked", G_CALLBACK(button_
clicked), NULL);
  g_signal_connect_swapped(G_OBJECT(button), "clicked", G_
CALLBACK(gtk_widget_destroy), G_OBJECT(window));
  gtk_container_add(GTK_CONTAINER(window), button);
  gtk_widget_show(button);
  gtk_widget_show(window);
  gtk_main();
  return 0;
}
```

Внимательный читатель сразу заметит, что мы добавили новый заголовочный файл – **libintl.h**. Этот файл содержит объявления функций, макросов и прочего, относящегося к интернационализации *GNU gettext*. Заголовочный файл **libintl.h** соответствует библиотеке *libintl*, которая должна быть скомпонована с нашей программой. Эта библиотека является частью *glibc*, а потому, если сборка программы выполняется в Linux (или другой системе, использующей *glibc*), связывание происходит автоматически и никаких дополнительных ключей не требуется. Вслед за директивами #include мы объявляем три макроса и две константы. Эти макросы представляют собой стандартную часть любого приложения, использующего интернационализацию GNU. Макрос _() является псевдонимом макроса gettext(). Сам макрос gettext() выполняет две функции, о которых говорилось в теоретическом введении – помечает в исходном тексте программы те строки текста, которые требуют перевода, и заменяет оригинальную строку ее переводом во время выполнения программы. Мы используем псевдоним ради удобства, поскольку печатать один символ «_» проще, чем вводить слово *gettext*. Забегая вперед, отметим, что с помощью макроса _() мы помечаем для перевода две строки – *Hello World!* и *Quit*. Константа GETTEXT_PACKAGE указывает общее имя файлов каталогов сообщений данного приложения, из которых программа должна будет брать переводы строк. Файлов каталогов сообщений у каждого приложения может быть много (по одному файлу для каждого языка и кодировки, на которые переведен интерфейс приложения), но все они имеют одно и то же имя (обычно соответствующее имени приложения, с добавлением расширения **.mo**). При этом никакого конфликта не возникает, поскольку файлы, соответствующие разным языкам, хранятся в разных поддиректориях корневой директории каталогов сообщений. Если вы откроете одну из корневых директорий, в которой по умолчанию хранятся ресурсы локализации, например, **/usr/share/locale**, то увидите в ней множество поддиректорий, имена которых совпадают с сокращенными именами различных локалей. В каждой из этих директорий вы найдете поддиректорию **LC_MESSAGES**. В ней обычно хранятся файлы локализации различных приложений. В системе есть несколько директорий, используемых для хранения файлов переводов. Программы GNOME ищут каталоги сообщений в поддиректориях **/opt/gnome/share/locale**. Универсальным хранилищем файлов переводов для разных приложений, использующих интернационализацию, основанную на *GNU gettext*, служит директория **/usr/share/locale**. Очевидно, что эти директории уместно использовать для хранения ресурсов тех приложений, которые установлены в системе глобально и доступны всем пользователям. Константа LOCALEDIR позволяет нам указать нестандартную корневую директорию для хранения файлов локализации нашего приложения. В качестве таковой мы указываем директорию locale, которая должна располагаться в рабочей директории программы.

В начале функции main() мы вызываем две функции, загружающие и настраивающие ресурсы локализации. Функция bindtextdomain(3) указывает системе интернационализации имя файлов ресурсов локализации и имя корневой директории, в которой они хранится. Функция bind_textdomain_codeset(3) позволяет указать кодировку переведенных сообщений в файлах локализации. Во время выполнения наша программа определит имя локали, в которой она работает, и будет искать файл с именем, заданным GETTEXT_PACKAGE, в соответствующей директории. Как видите, все английские строки, требующие пере- »

вода, представлены в программе как аргументы макроса _() (который, напомню, является синонимом макроса gettext()).

На этом этапе интерфейс программы *helloworld* готов к переводу на другие языки. Наша следующая задача – извлечь из исходного текста программы список строк, которые надлежит перевести. Это делается с помощью уже упомянутой утилиты *xgettext*. В окне консоли даем команду:

```
xgettext -C helloworld.c -k_
```

Ключ –C указывает программе, что она имеет дело с исходным файлом C/C++. Ключ –k позволяет нам указать вид маркера, которым помечены строки для перевода. В нашем случае маркером служит знак подчеркивания. Утилита *xgettext* не вносит никаких изменений в файл **helloworld.c**, но в результате ее выполнения на диске появится файл **messages.po**, содержащий все строки из файла **helloworld.c**, помеченные макросом _(). Мы сделаем копию этого файла и назовем ее **ru.po**, а затем добавим в нее русский перевод строк интерфейса. Файлы переводов ***.po** представляют собой документы XML. Для работы с файлами переводов можно воспользоваться редактором *Emacs*, можно даже добавлять строки перевода «вручную» в обычном текстовом редакторе (при этом, конечно, необходимо соблюдать формат файлов ***.po**). Однако в вашей системе наверняка установлен гораздо более удобный инструмент, который, правда, не является частью пакетов *GTK+* или *GNU Gettext*. Речь идет о редакторе файлов перевода *KBabel*, входящем в состав пакета разработчика KDE (**рис. 1**).

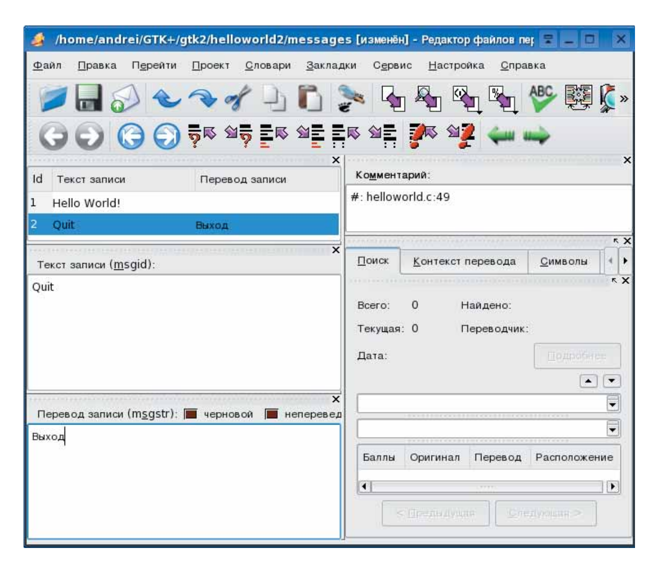


Рис. 1. *KBabel* в работе.

Окно *KBabel* вертикально разделено на две половины. Левая половина содержит три окна. В самом верхнем окне вы найдете список всех строк, подлежащих переводу. В среднем окне можно выбрать строку для перевода, а в нижнем окне – ввести сам перевод. Правая часть окна *KBabel* содержит некоторые вспомогательные инструменты. Выполним перевод всех строк на русский язык и сохраним изменения, внесенные в файл **ru.po**. Если вы просматриваете пиктограмму файла **ru.po** в менеджере *Konqueror*, то можете заметить, что цвет круга на пиктограмме файла изменился с красного на зеленый. Это значит, что все содержимое файла переведено (круг на иконке файла представляет собой диаграмму, отображающую процент переведенных строк). Полученный нами файл перевода **ru.po** представляет собой «исходник» двоичного ресурса локализации. Для того чтобы скомпилировать его в «машинное» представление, мы воспользуемся утилитой *msgfmt*:

```
msgfmt ru.po -o helloworld.mo
```

В результате появится файл **helloworld.mo**, который мы сможем распространять вместе с двоичной версией нашей программы. Для того, чтобы во время выполнения программа могла использовать этот файл, его нужно поместить туда, где программа ожидает его найти. В нашем случае это директория **./locale/ru/LC_MESSAGES**. Нам, кажется, осталось только скомпилировать программу *helloworld*. Теперь при ее запуске вы увидите надписи на русском языке (**рис. 2**).

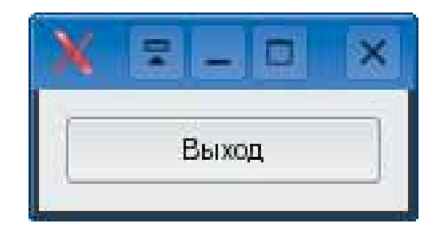


Рис. 2. Русифицированная программа helloworld.

В нашем кратком обзоре мы, разумеется, рассмотрели далеко не все аспекты интернационализации и локализации приложений с помощью системы *GNU gettext*, так что пренебрегать учебником GNU ни в коем случае не следует. Отметим здесь еще одну утилиту, которая может оказаться полезной в процессе локализации сложных проектов. Представьте себе, что вы локализуете свою (или чужую) программу. Вы создали файл каталога сообщений, перевели все сообщения на ваш родной (или не родной) язык, скомпилировали двоичный каталог сообщений... Но жизнь не стоит на месте, и программа, локализацией которой вы занимаетесь, продолжает развиваться. В ней появляются новые строки, требующие перевода. Было бы очень неразумно начинать весь процесс локализации сначала из-за того, что в программе появилась новая текстовая строка. Ведь у вас уже есть каталог сообщений, который содержит перевод всех остальных строк. Если вы не хотите переписывать вручную весь перевод каждый раз, когда каталог сообщений, генерируемый утилитой *xgettext*, изменится, вам на помощь придет утилита *msgmerge*. Эта утилита позволяет объединить старый, уже переведенный каталог сообщений и новый каталог, содержащий дополнения.

Две кнопки

Для того, чтобы исследовать возможности *GTK+*, нам, конечно, понадобятся программы с более сложным интерфейсом, чем окно с одной кнопкой, так что сейчас будет логично рассмотреть некоторые принципы построения интерфейсов программ *GTK+*. В первой статье этой серии мы уже упоминали объекты-контейнеры, которые управляют размером и расположением дочерних визуальных элементов. Главное окно приложения *GTK+* само представляет собой объект-контейнер. Впрочем, возможности главного окна как объекта-контейнера весьма ограничены. В частности, попытка добавить в окно приложения вторую кнопку наталкивается на неожиданное препятствие, – у главного окна приложения может быть только один дочерний визуальный элемент. Если мы хотим, чтобы окно приложения содержало более одного визуального элемента (естественное желание, не правда ли?), мы должны сначала разместить в окне дочерний объект-контейнер, способный управлять большим числом элементов. Объектов-контейнеров в *GTK+* реализовано немало. Все они являются потомками объекта GtkContainer, реализующего абстрактный контейнер. В частности, объект-контейнер главного окна принадлежит классу GtkBin, представляющему собой контейнер, способный содержать (вы догадались!) только один дочерний элемент. Другие контейнеры позволяют управлять сразу многими дочерними визуальными элементами.

Все контейнеры *GTK+* можно разделить на две категории. Одни контейнеры управляют расположением дочерних визуальных элементов, но сами визуальными элементами не являются. К этой категории контейнеров относится контейнер GtkHBox, с которым мы познакомимся ниже. Ко второй категории относятся контейнеры, которые включают определенные визуальные элементы управления «по умолчанию». Примером контейнеров этого типа может служить контейнер GtkNotebook, который создает панель с несколькими вкладками.

Рассмотрим пример использования простого контейнера GtkHBox, который позволяет расположить несколько дочерних элементов гори-

зонтально. Ниже приведен текст программы *buttontest*, в которой мы пользуемся не одной, а двумя кнопками.

```
#include <gtk/gtk.h>
static void button_clicked(GtkWidget * widget, gpointer data)
{
  gint i = * (gint *) data;
  g_print("Button #%i is pressed!\n", i);
}
static gboolean delete_event(GtkWidget * widget, GdkEvent * event, gpointer data)
{
  g_print("Delete event occurred\n");
  return FALSE;
}
static void destroy(GtkWidget * widget, gpointer data)
{
  g_print("Destroy signal sent\n");
  gtk_main_quit();
}
int main(int argc, char ** argv)
{
  GtkWidget * window;
  GtkWidget * button1;
  GtkWidget * button2;
  GtkWidget * box;
  gint i1, i2;
  gtk_init(&argc, &argv);
  window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
  gtk_window_set_title(GTK_WINDOW(window), "Buttons");
  gtk_container_set_border_width(GTK_CONTAINER(window), 10);
  g_signal_connect(G_OBJECT(window), "delete_event", G_CALLBACK(delete_event), NULL);
  g_signal_connect(G_OBJECT(window), "destroy", G_CALLBACK(destroy), NULL);
  box = gtk_hbox_new(TRUE, 2);
  gtk_container_add(GTK_CONTAINER(window), box);
  button1 = gtk_button_new_with_label("Выход");
  i1 = 1;
  g_signal_connect(G_OBJECT(button1), "clicked", G_CALLBACK(button_clicked), &i1);
  g_signal_connect_swapped(G_OBJECT(button1), "clicked", G_CALLBACK(gtk_widget_destroy), G_OBJECT(window));
  gtk_box_pack_start(GTK_BOX(box), button1, TRUE, TRUE, 0);
  button2 = gtk_button_new_with_label("Кнопка 2");
  i2 = 2;
  g_signal_connect(G_OBJECT(button2), "clicked", G_CALLBACK(button_clicked), &i2);
  gtk_box_pack_start(GTK_BOX(box), button2, TRUE, TRUE, 0);
  gtk_widget_show(button1);
  gtk_widget_show(button2);
  gtk_widget_show(box);
  gtk_widget_show(window);
  gtk_main();
  return 0;
}
```

В этом примере дочерним элементом главного окна программы содержит контейнер box, который, в свою очередь, включает две кнопки – button1 и button2. Контейнер типа GtkHBox создается функцией-конструктором gtk_hbox_new(). Конструктор принимает два аргумента. Первый аргумент, значение типа gboolean, позволяет указать, должны ли дочерние элементы контейнера иметь одинаковые размеры. Если передать в этом аргументе значение **true**, размеры дочерних элементов будут одинаковыми. Второй аргумент имеет тип gint и позволяет указать расстояние между дочерними элементами контейнера в пикселях. Здесь уместно еще раз обратить внимание на систему типов *GTK+*. В целях улучшения переносимости с одной платформы на другую, *GTK+* определяет собственные аналоги простых типов данных C. Многие из этих типов представляют собой псевдонимы типов C со схожими именами. Например, тип gint, является псевдонимом типа int. Сложнее обстоит дело с типом gboolean. В языке C (в отличие от C++) нет встроенного булевого типа, и тип gboolean является псевдонимом типа gint. Константы TRUE и FALSE объявлены так, чтобы соответствовать результатам логических операций C. (FALSE **= 0**, TRUE **= !FALSE**, то есть любое ненулевое значение).

После того, как мы создали контейнер, мы делаем его дочерним элементом главного окна с помощью знакомой нам функции gtk_container_add(). Далее мы создаем кнопку и назначаем ей обработчики сигналов, так же, как мы делали в предыдущем примере. Наша следующая задача – добавить кнопку в контейнер box. Для этого мы воспользуемся функцией gtk_box_pack_start(). Функция gtk_box_pack_start() добавляет новые визуальные элементы в контейнер в порядке слева направо, если контейнер представляет собой «горизонтальный ящик» GtkHBox, и сверху вниз, если контейнер представляет собой «вертикальный ящик» GtkVBox. Функция gtk_box_pack_end() добавляет элементы в противоположном порядке, соответственно справа налево и снизу вверх. Функции gtk_box_pack_start() и gtk_box_pack_end() обладают одинаковым набором параметров. Первым параметром каждой функции является указатель на объект-контейнер. Вторым параметром служит указатель на добавляемый объект. Третий и четвертый параметры позволяют указать порядок добавления и распределения нового пространства, добавляемого вместе с новым элементом. Последний параметр указывает, сколько дополнительных пикселей следует расположить между новым элементом и его соседями.

Хотя объект box сам по себе не создает никаких визуальных элементов, его следует сделать видимым с помощью функции gtk_widget_show(), как и все визуальные элементы программы. В результате получаем окно с двумя кнопками (**рис. 3**).

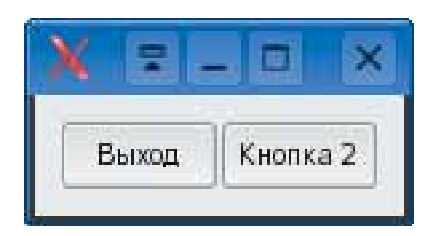


Рис. 3. Окно с двумя кнопками.

Вооружившись познаниями о контейнерах, мы сможем свободно исследовать другие визуальные элементы *GTK+*, а также подробнее изучить механизм сигналов, чему и будет посвящена следующая статья. LXF

Через месяц Мы займемся углубленным изучением механизма сигналов GTK+.

Потоки:

ЧАСТЬ 8: Подобно леммингам, бесконтрольно размножающиеся потоки очень быстро устраивают гонку-соревнование за ресурсы системы. Но **Андрей Боровский** знает, как умерить их аппетиты...



Мы продолжаем знакомство с многопоточными приложениями Linux. В предыдущей статье мы научились создавать потоки и вызывать их досрочное завершение. Мы уже знаем, что если запрос на досрочное завершение потока поступил «в неподходящий момент», поток может повременить с кончиной до тех пор, пока не станет готов к ней. Механизм отложенного досрочного завершения очень полезен, но для действительно эффективного управления завершением потоков необходим еще и механизм, оповещающий поток о досрочном завершении. Оповещение о завершении потоков в Unix-системах реализовано на основе тех же принципов, что и оповещение о завершении самостоятельных процессов. Если нам нужно выполнять какие-то специальные действия в момент завершения потока, мы устанавливаем функцию-обработчик, которая будет вызвана перед тем, как поток завершит свою работу. Смысл назначения обработчика заключается в том, что он будет вызван как при нормальном завершении потока, так и при досрочном завершении. Для потоков наличие обработчика завершения даже более важно, чем для процессов. Предположим, что поток выделяет блок динамической памяти и затем внезапно завершается по требованию другого потока. Если бы поток был самостоятельным процессом, ничего особенно неприятного не случилось бы, так как ОС сама убрала бы за ним мусор. В случае же процесса-потока невысвобожденный блок памяти так и останется «висеть» в адресном пространстве многопоточного приложения. Если потоков много, а ситуации, требующие досрочного завершения, возникают часто, утечки памяти могут оказаться значительными. Устанавливая обработчик завершения потока, высвобождающий занятую память, мы можем быть уверены, что поток не оставит за собой «бесхозных» блоков (если, конечно, в системе не случится какого-то более серьезного сбоя).

Для установки обработчика завершения потока применяется макрос pthread_cleanup_push(3). Подчеркиваю жирной красной чертой, pthread_cleanup_push() – это макрос, а не функция. Неправильное использование данного макроса может привести к неожиданным синтаксическим ошибкам.

У pthread_cleanup_push() два аргумента. В первом передается адрес функции-обработчика завершения потока, а во втором – нетипизированный указатель, который будет передан как аргумент функции-обработчику. Этот указатель может указывать на что угодно – мы сами решаем, какие данные должны быть переданы обработчику завершения потока. Макрос pthread_cleanup_push() помещает переданные ему адрес функции-обработчика и указатель в специальный стек. Само слово «стек» указывает, что мы можем назначить потоку произвольное число функций-обработчиков завершения. Поскольку в стек записывается не только функция, но и ее аргумент, мы можем назначить один и тот же обработчик с несколькими разными аргументами.

Для того, чтобы обработчики смогли выполнить свою задачу, они, естественно, должны быть вызваны в подходящий момент. В процессе завершения потока функции-обработчики и их аргументы должны быть извлечены из стека и выполнены. Извлечение обработчиков из стека с последующим выполнением может быть выполнено либо явно, либо автоматически. Автоматически обработчики завершения потока выполняются при вызове потоком функции pthread_exit() (которая завершает его работу), а также при выполнении потоком запроса на досрочное завершение. Явным образом обработчики завершения потока извлекаются из стека с помощью макроса pthread_cleanup_pop(3). Во всех случаях обработчики извлекаются из стека и выполняются в порядке, противоположном тому, в котором они были помещены в стек. Если мы используем макрос pthread_cleanup_pop() явно, мы можем указать, что обработчик необходимо только извлечь из стека, но выполнять его не следует. Порядок назначения и выполнения обработчиков выглядит довольно сложным, поэтому мы начнем его изучение с простого примера, программы *exittest*:

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <pthread.h>
```

Месяц назад Мы узнали как создавать и принудительно завершать потоки.



синхронизация

```
void exit_func(void * arg)
{
  free(arg);
  printf("Freed the allocated memory.\n");
}
void * thread_func(void *arg)
{
  int i;
  void * mem;
  pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, NULL);
  mem = malloc(1024);
  printf("Allocated some memory.\n");
  pthread_cleanup_push(exit_func, mem);
  pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, NULL);
  for (i = 0; i < 4; i++) {
    sleep(1);
    printf("I'm still running!!!\n");
  }
  pthread_cleanup_pop(1);
}
int main(int argc, char * argv[])
{
  pthread_t thread;
  pthread_create(&thread, NULL, thread_func, NULL);
  pthread_cancel(thread);
  pthread_join(thread, NULL);
  printf("Done.\n");
  return EXIT_SUCCESS;
}
```

В этой программе (вы найдете ее на диске в файле **exittest.c**) много уже знакомых нам элементов. Программа *exittest* создает дополнительный поток и тут же посылает запрос на его завершение. Новые для нас элементы сосредоточены в функции потока thread_func(). Поток начинает работу с того, что запрещает досрочное завершение. Этот запрет необходим на время выполнения важных действий, которые нельзя прерывать. Если запрос поступит во время действия запрета, он не пропадет. Как мы уже знаем, запрет досрочного завершения не отменяет выполнение запроса на досрочное завершение, а откладывает его. Далее поток динамически выделяет блок памяти. Чтобы избежать утечек памяти, мы должны гарантировать высвобождение выделенного блока. Задачу высвобождения блока памяти мы возлагаем на функцию exit_func(), которая является обработчиком завершения потока. Для этого мы добавляем функцию exit_func() в стек обработчиков завершения потока с помощью макроса pthread_cleanup_push().

Обратите внимание на второй параметр макроса. Им, как мы знаем, должен быть нетипизированный указатель. Этот указатель будет передан в качестве аргумента функции-обработчику. Поскольку задача функции exit_func() заключается в том, чтобы высвободить блок памяти mem, в качестве аргумента функции мы просто передаем указатель на этот блок. Функция exit_func() высвобождает блок памяти с помощью free(3) и выводит диагностическое сообщение.

Продолжим изучение функции потока thread_func(). После установки обработчика exit_func() наш поток разрешает досрочное завершение. Теперь при поступлении запроса на досрочное завершение блок памяти mem будет высвобождаться автоматически. Далее поток выводит четыре диагностических сообщения с интервалом в одну секунду и завершает свою работу. Перед выходом из функции потока мы вызываем макрос pthread_cleanup_pop(). Этот макрос извлекает функцию-обработчик из стека. Аргумент макроса позволяет указать, следует ли выполнять функцию-обработчик, или требуется только удалить ее из стека. Мы передаем макросу ненулевое значение, что указывает на необходимость выполнить обработчик.

Если вы забудете поставить вызов pthread_cleanup_pop() в конце функции потока, компилятор выдаст сообщение о синтаксической ошибке. Объясняется это, конечно, тем, что pthread_cleanup_push() и pthread_cleanup_pop() – макросы. Первый макрос, кроме прочего, открывает фигурные скобки, которые второй макрос должен закрыть, так что число обращений к pthread_cleanup_push() в функции потока всегда должно быть равно числу обращений к pthread_cleanup_pop(), иначе программу не удастся откомпилировать.

Интересен вопрос о взаимодействии вызовов pthread_cleanup_pop() и pthread_exit(). Мы уже говорили, что вызов pthread_exit() приводит к опустошению стека и последовательному выполнению всех обработчиков. Означает ли это, что если функция потока завершает с помощью pthread_exit(), то вызывать макросы pthread_cleanup_pop() уже не нужно? Нет, не означает. Ведь макросы, как мы уже видели, управляют синтаксической структурой программы на этапе компиляции, и вызов pthread_exit() не может заменить их в этой роли. Мы можем расположить вызов pthread_exit() до вызовов pthread_cleanup_pop() (в этом случае поток завершится до обращения к макросам, но поскольку pthread_exit() сама опустошает стек обработчиков, этого уже и не требуется). Мы также можем расположить вызов pthread_exit() после вызовов pthread_cleanup_pop(), в этом случае стек обработчиков будет опустошен до вызова pthread_exit() и эта функция просто завершит работу программы. Тогда возникает другой вопрос: а нужно ли вообще вызывать pthread_exit() в конце функции потока, если вызовы макросов pthread_cleanup_pop() все равно необходимы? Ответ на него зависит от обстоятельств. Помимо вызова обработчиков завершения, функция pthread_exit() может выполнять в вашем потоке и другие действия финализации, и в этом случае ее вызов необходим. Еще один тонкий момент связан с выходом из функции потока с помощью оператора return. Сам по себе return не приводит к вызову обработчиков завершения. В нашем примере мы вызвали обработчик явно с помощью pthread_cleanup_pop(), но рассмотрим такой вариант функции thread_func():

```
void * thread_func(void *arg)
{
  int i;
  void * mem;
  pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, NULL);
  mem = malloc(1024);
  printf("Allocated some memory.\n");
  pthread_cleanup_push(exit_func, mem);
  pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, NULL);
  for (i = 0; i < 4; i++) {
```

```
  sleep(1);
  printf("I'm still running!!!\n");
  if (i == 2) return;
 }
 pthread_cleanup_pop(1);
 pthread_exit(0)
}
```

Пусть этот вариант выглядит несколько неестественным, но верно отражает суть: теперь в функции потока определено несколько точек выхода. В приведенном выше варианте thread_func() завершится вызовом return, и обработчик завершения потока при этом вызван не будет. Вариант

```
...
if (i == 2) {
 pthread_cleanup_pop(1);
 return;
}
...
```



вообще не скомпилируется, поскольку «лишний» макрос pthread_cleanup_pop() нарушит синтаксис программы. Правильный ответ состоит в использовании pthread_exit():

```
if (i == 2) pthread_exit(0);
```

Вполне возможно, что вам, уважаемый читатель, как и мне, уже несколько раз хотелось досрочно завершить обсуждение досрочного завершения потоков. Потерпите немного, мы уже приближаемся к финишу. Осталось лишь ответить не вопрос, зачем нам нужна возможность устанавливать несколько обработчиков завершения потока? Ответов на этот вопрос может быть много, но я дам только один. Представьте себе, что вы программируете сложную функцию потока, которая интенсивно работает с динамической памятью. Как только в вашей функции выделяется новый блок памяти, вы устанавливаете обработчик завершения потока, который высвободит его в случае неожиданного завершения. Тут стоит отвлечься на секунду и заметить, что установка обработчика, высвобождающего память во время завершения потока, не мешает вам самостоятельно высвободить эту память, когда она перестанет быть нужна. Придется только немного поиграть с указателями (на диске вы найдете программу **exittest2.c**, которая демонстрирует явное высвобождение памяти в потоке совместно с использованием обработчика завершения). Если затем в вашей функции понадобится выделить новый блок памяти, потребуется еще один обработчик для его высвобождения. Даже если вы заранее знаете, сколько раз ваша программа будет выделать блоки памяти, назначать обработчик для высвобождения каждого блока можно только после того, как он был выделен.

Средства синхронизации потоков

Изучая взаимодействие между процессами, мы уделили большое внимание средствам синхронизации. У потоков тоже есть для этого свои специальные механизмы. Вернемся к первому примеру из LXF86, программе *threads*. Напомню, что в том примере мы создавали два потока, используя одну и ту же функцию thread_func. В процессе создания каждого потока этой функции передавалось целочисленное значение (номер потока). При этом для передачи значения каждому потоку использовалась своя переменная. В той статье я подробно объяснил, почему использование одной и той же переменной для передачи значения функциям разных потоков может привести к ошибке. Коротко говоря – проблема заключалась в том, что мы не могли знать, когда именно новый поток начнет свое выполнение. С помощью средств синхронизации потоков мы можем решить эту проблему и использовать одну переменную для передачи значений обоим потокам. Рассмотрим модифицированный вариант программы *threads* – *threads2* (вы найдете ее на диске в файле **threads2.c**).

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <pthread.h>
#include <semaphore.h>
sem_t sem;
void * thread_func(void *arg)
{
 int i;
 int loc_id = * (int *) arg;
 sem_post(&sem);
 for (i = 0; i < 4; i++) {
  printf("Thread %i is running\n", loc_id);
  sleep(1);
 }
}
int main(int argc, char * argv[])
{
 int id, result;
 pthread_t thread1, thread2;
 id = 1;
 sem_init(&sem, 0, 0);
 result = pthread_create(&thread1, NULL, thread_func, &id);
 if (result != 0) {
  perror("Creating the first thread");
  return EXIT_FAILURE;
 }
 sem_wait(&sem);
 id = 2;
 result = pthread_create(&thread2, NULL, thread_func, &id);
 if (result != 0) {
  perror("Creating the second thread");
  return EXIT_FAILURE;
 }
 result = pthread_join(thread1, NULL);
 if (result != 0) {
  perror("Joining the first thread");
  return EXIT_FAILURE;
 }
 result = pthread_join(thread2, NULL);
 if (result != 0) {
  perror("Joining the second thread");
  return EXIT_FAILURE;
 }
 sem_destroy(&sem);
 printf("Done\n");
 return EXIT_SUCCESS;
}
```

Как видите, теперь мы используем одну переменную id для передачи значения обоим потокам. Если вы скомпилируете и запустите программу, то увидите, что она работает корректно. Секрет нашего успеха заключается в использовании средств синхронизации, а именно – семафоров. Мы уже знакомились с семафорами System V в LXF82 – тогда они использовались для синхронизации процессов. В данном случае мы имеем дело с семафорами другого типа – семафорами POSIX, которые специально предназначены для работы с потоками. Все объявления функций и типов, относящиеся к этим семафорам, можно найти в файле **/usr/include/nptl/semaphore.h**. Семафоры создаются (инициализируются) с помощью функции sem_init(3). Первый параметр функции sem_init() – указатель на переменную типа sem_t, которая служит идентификатором семафора. Второй параметр, pshared, в настоящее время не используется. Мы оставим его равным нулю. В третьем параметре передается значение, которым инициализируется семафор. Дальнейшая работа с семафором осуществляется с помощью функций sem_wait(3) и sem_post(3). Единственным аргументом функции sem_wait() служит указатель на идентификатор семафора. Функция sem_wait() приостанавливает выполнение вызвавшего ее потока до тех пор, пока значение семафора не станет больше нуля, затем функция уменьшает значение семафора на единицу и возвращает управление. Функция sem_post(), наоборот, увеличивает значение семафора, иден-

тификатор которого был ей передан, на единицу. Присвоив семафору значение 0, наша программа создает первый поток и вызывает функцию sem_wait(). Эта функция вернет управление программе после того, как функция потока вызовет функцию sem_post(), а это случится не раньше, чем thread_func() обработает значение id. Таким образом, мы можем быть уверены, что когда sem_wait() вернет управление функции main, первый поток уже закончит обработку переменной id, и мы сможем использовать эту переменную для передачи данных следующему потоку. После завершения обоих потоков семафор перестает быть нужным, и мы можем вызвать функцию sem_destroy(3) для его удаления и высвобождения ресурсов.

Семафоры – не единственное средство синхронизации потоков. Для разграничения доступа к глобальным объектам потоки могут использовать мьютексы [mutex, от английского **mutual exclusive** - взаимоисключающий, – прим. ред.]. Все функции и типы данных, имеющие отношение к мьютексам, определены в файле **pthread.h**. Мьютекс создается вызовом функции pthread_mutex_init(3). В качестве первого аргумента этой функции передается указатель на переменную pthread_mutex_t. Она играет роль идентификатора создаваемого мьютекса. Вторым аргументом функции pthread_mutex_init() должен быть указатель на переменную типа pthread_mutexattr_t. Эта переменная позволяет установить дополнительные атрибуты мьютекса. Если нам нужен обычный мьютекс, мы можем передать во втором параметре **NULL**. Для того, чтобы получить исключительный доступ к некому глобальному ресурсу, поток вызывает функцию pthread_mutex_lock(3) (в этом случае говорят, что «поток захватывает мьютекс»). Единственным аргументом функции pthread_mutex_lock() является идентификатор мьютекса. Закончив работу с глобальным ресурсом, поток высвобождает мьютекс с помощью функции pthread_mutex_unlock(3). Если поток вызовет функцию pthread_mutex_lock() для мьютекса, уже захваченного другим потоком, эта функция не вернет управление до тех пор, пока другой поток не высвободит мьютекс с помощью вызова pthread_mutex_unlock(). Если мьютекс больше не нужен, его можно удалить из системы с помощью функции pthread_mutex_destroy(3). Стоит отметить, что в отличие от многих других функций, приостанавливающих работу потока, вызов pthread_mutex_lock() не является точкой останова. Иначе говоря, поток, находящийся в режиме отложенного досрочного завершения, не может быть завершен в тот момент, когда он ожидает выхода из pthread_mutex_lock().

Атрибуты потоков

Создавая новый поток, вы можете указать ряд дополнительных атрибутов, определяющих некоторые параметры. Из всех этих атрибутов более всего востребован атрибут DETACHED, позволяющий создавать отделенные потоки. Во всех рассмотренных выше примерах мы использовали функцию pthread_join(), позволяющую дождаться завершения потока и получить значение, возвращенное его функцией. Для того, чтобы функция pthread_join() могла получить значение функции потока, завершившегося до вызова pthread_join(), система сохраняет данные о потоке после его завершения (это напоминает появление «зомби» после завершения самостоятельного процесса). Если наша программа интенсивно работает с потоками и синхронизация потоков с помощью pthread_join() нам не нужна, мы можем сэкономить ресурсы системы, используя отделенные потоки. Отделенные потоки отличаются от обычных (присоединяемых) потоков тем, что после завершения отделенного потока система не сохраняет о нем никакой информации. При попытке использовать для отделенного потока функцию pthread_join(), последняя вернет сообщение об ошибке.

Вы можете сделать поток отделенным с помощью вызова функции pthread_detach(3). Однако то же самое можно сделать с помощью атрибутов потоков, которыми мы и займемся. Для того, чтобы назначить потоку дополнительные атрибуты, нужно сначала создать объект, содержащий набор атрибутов. Этот объект создается функцией pthread_attr_init(3). Единственный аргумент этой функции – указатель на переменную типа pthread_attr_t, которая служит идентификатором набора атрибутов. Функция pthread_attr_init() инициализирует набор атрибутов потока значениями, заданными по умолчанию, так что мы можем работать только с теми атрибутами, которые нас интересуют, и не беспокоиться об остальных. Для добавления атрибутов в набор используются специальные функции с именами pthread_attr_set<имя_атрибута>. Например, для того, чтобы добавить атрибут «отделенности», мы вызываем функцию pthread_attr_setdetachstate(3). Первым аргументом этой функции должен быть адрес объекта набора атрибутов, а вторым аргументом – константа, определяющая значение атрибута. Константа PTHREAD_CREATE_DETACHED указывает, что создаваемый поток должен быть отделенным, тогда как константа PTHREAD_CREATE_JOINABLE определяет создание присоединяемого (joinable) потока, который может быть синхронизирован функцией pthread_join(3). После того, как мы добавили необходимые значения в набор атрибутов потока, мы вызываем функцию создания потока pthread_create(). Набор атрибутов потока передается ей в качестве второго аргумента.

На этом мы закончим ликбез, посвященный потокам. Если вы любите потоки также сильно, как люблю их я, к вашим услугам неисчерпаемое море специальной литературы, в которой вы найдете все то, чему не хватило места на страницах журнала. Ну а перед тем, как приступать к чтению следующей статьи этой серии, запаситесь серой, ибо мы познакомимся с демонами. LXF



Через месяц Быть здесь драконам... нет... демонам.

Потоки

ЧАСТЬ 4: Завершая курс молодого Java-бойца, **Антон Черноусов** научит вас управлять потоками... Жаль, что не денежными.



С каждым днем появляются все более мощные процессоры, многоядерная архитектура которых стала основной темой ушедшего года, поэтому двухядерный процессор в ноутбуке уже никого не удивляет. С одной стороны – это обстоятельство приближает возможности простого пользователя к возможностям «по-настоящему» больших систем. С другой (и рекламные буклеты об этом обычно молчат) – для того, чтобы использовать весь потенциал современных компьютеров, приложение должно «уметь» просчитать задачу фактически на двух или более процессорах.

Создание эффективных алгоритмов для работы на многопроцессорных станциях – это большая и сложная работа. Несмотря на это, для любого программиста актуальна задача организации взаимодействия с медленными ресурсами (например, чтения, записи или копирования файлов, работы с принтером, сетью), так как немногие пользователи смирятся с тем, что их любимая программа «замирает» в момент выполнения какой-либо операции.

Во избежание описанных проблем программа должна использовать потоки или процессы. Под процессом понимается заявка на потребление всех видов ресурсов системы, кроме одного – процессорного времени, или иначе говоря, процесс – это запущенная на выполнение программа (такое определение дается в [1]). Поток рассматривается как самостоятельная активность внутри процесса, хотя существуют другие трактовки этого понятия, которые зависят от используемой операционной системы (см., например, [2]). Поток получил свое название по аналогии с потоком команд, поступающих в процессор; при выполнении потоки делят адресное пространство и выделенную память внутри одного процесса. Процессорное время распределяется между различными потоками операционной системой, точнее, одним из компонентов ее ядра – планировщиком. Более полно с понятием процессов и потоков и механизмов работы с ними с точки зрения операционной системы вы можете ознакомиться в книге [3].

Давайте завершим наш экскурс в теорию и окунемся в реальность Java. Под процессом здесь принято понимать всеобъемлющий контекст выполнения, обеспечивающий высокий уровень изоляции охватываемых им данных от внешнего мира, а под потоком – более «легковесный» активный агент; в контексте одного процесса может функционировать целое множество потоков [4]. Планирование потоков в Java обеспечивается внутренними механизмами JVM.

Поток, он же thread

В Java существует два способа работы с потоками: первый заключается в реализации интерфейса Runnable, второй связан с наследованием класса Thread, который уже реализует данный интерфейс. В обоих случаях класс должен предоставлять реализацию метода run(). Ниже приведен пример класса, реализующего поток через наследование класса Thread:

```
public class FirstThread extends Thread {
 public void run(){
  for (int i = 1 ; i < 30; i++)
        System.out.println("It is in thread "+ i);
 }
}
```

Собственно, метод run() и должен содержать некоторый набор инструкций (разумеется, на языке Java), которые вы хотите выполнить в отдельном потоке. Например, если вы реализуете функцию копирования файла (а пользователь, скажем, копирует ISO-образ объемом 600 Мб), желательно, чтобы эта операция выполнялась в отдельном потоке, запуск которого можно производить следующим образом:

```
public class ConsoleToThread {
 public static void main(String[] args) {
  FirstThread thread = new FirstThread();
  thread.start();
  for (int i = 1; i < 20; i++) {
   System.out.println("It is in main " + i);
  }
```

Литература

1. П. Кью «**Использование UNIX**», ISBN 5-8275-0019-4
2. В.Г. Олифер, Н.А. Олифер «**Сетевые операционные системы**», ISBN 5-272-00120-6
3. Д. Бэкон, Т. Харрис «**Операционные системы**», ISBN 5-94723-969-8
4. М. Фаулер «**Архитектура корпоративных программных приложений**», ISBN 5-8459-0579-6

Месяц назад Мы научились работать с файлами и документами XML

в Java

```
  try {
    thread.join();
  } catch (InterruptedException ex) {
    System.out.println("Exception in stop thread");
  }
 }
}
```

При выполнении метода main() класса ConsoleToThread создается объект-поток FirstThread, который запускается на выполнение методом start() [заметьте – метод run() никогда не вызывается явно, – прим. ред.]. Метод join() используется в случае, когда необходимо «дождаться» завершения потока. Завершение работы потока происходит при выходе из метода run(), как явном (например, посредством return), так и неявном (если внутри метода возникло и не было обработано какое-то исключение).

Имейте в виду (это важно!): повторный запуск уже отработавшего потока приведет к исключению IllegalThreadStateException.

Реализация потока через Runnable

Давайте теперь рассмотрим пример работы с потоками через интерфейс Runnable. Если, допустим, класс SameRunnable реализует интерфейс Runnable, то запустить поток на основе этого класса на выполнение можно следующим образом:

```
 Runnable run = new SameRunnable();
Thread thread = new Thread(run);
thread.start();
```

В следующем примере в методе main() класса ConsoleToThreadTwo создается массив threadArray, состоящий из объектов-потоков. При этом используется конструктор класса Thread, принимающий два параметра: ссылку на объект, реализующий интерфейс Runnable и имя потока:

```
 Thread thread = new Thread(Runnable, ThreadName);
```

Метод getName() объекта, реализующего поток, возвращает указанное при создании имя. Обратите внимание, что в нем используется статический метод Thread.currentThread(), возвращающий ссылку на объект Thread, соответствующий выполняющемуся в текущий момент потоку:

```
public class SecondThread implements Runnable {
  public String getName() {
    return Thread.currentThread().getName();
  }

  public void run() {
    for (int i = 1; i < 100000; i++) {
      if ((i % 10000) == 0) {
        System.out.println(getName() + " counts " + i / 10000);
    }}}}

public class ConsoleToThreadTwo {
  public static void main(String[] args) {
    Thread[] threadArray = new Thread[3];
    for (int i=0; i<threadArray.length; i++){
      threadArray[i] = new Thread(new SecondThread(), "Thread " + i);
    }
    for (int i=0; i<threadArray.length; i++){
      threadArray[i].start();
      System.out.println(threadArray[i].getName() + " started");
    }}}
```

Проследив за выводом этой программы, можно заметить, что процессорное время распределяется между потоками практически равномерно, однако порядок их выполнения во многом случаен.

Приоритеты потоков

Для управления величиной процессорного времени, выделяемого потоку, можно воспользоваться приоритетами. Установка приоритетов происходит с помощью метода Thread.setPriority(), узнать текущий приоритет позволяет метод getPriority(). В классе Thread определены три константы:

```
· MIN_PRIORITY = 1
· NORM_PRIORITY = 5
· MAX_PRIORITY = 10
```

Важно понимать, что значение приоритета потока предназначено для Java-машины и не соответствует реальным приоритетам потоков в операционной системе.

Давайте немного изменим код метода main() класса ConsoleToThreadTwo:

```
  public static void main(String[] args) {
  Thread[] threadArray = new Thread[3];
    int pr = 0;
    for (int i=0; i<threadArray.length; i++){
    threadArray[i] = new Thread(new SecondThread(), "Thread " + i);
    if (pr == 10) pr = 0;
    threadArray[i].setPriority(Thread.MIN_PRIORITY + pr);
    pr++;
  }
  for (int i=0; i<threadArray.length; i++){
    threadArray[i].start();
    System.out.println(threadArray[i].getName() + " started");
  }}
```

Анализ результатов работы класса показывает, что потоки, получившие более высокий приоритет, выполняются чаще. Также, благодаря условию на значение переменной pr, setPriority() никогда не будет передан приоритет, превышающий 10 (MAX_PRIORITY). Если бы это произошло, система выбросила бы исключение IllegalArgumentException.

Потоки-демоны

Сделаем еще одно важное замечание: программа будет выполняться до тех пор, пока выполняется хотя бы один запущенный в ней поток; единственным исключением являются потоки-демоны.

Что же это такое? Демон отличается от «простого смертного» потока вызовом метода setDeamon(true), который необходимо сделать до начала работы. Например, так:

```
FirstThread thread = new FirstThread();
thread.setDeamon(true);
thread.start();
```

Узнать, является ли поток демоном, можно с помощью метода isDeamon(). Обычно потоки-демоны создаются для обслуживания некритичных задач, так как при завершении работы программа не дожидается их остановки, а прерывает их самостоятельно.

»



» Где искать потоки?

В большинстве случаев бывает необходимо отслеживать ранее запущенные на выполнение потоки. Использование массива потоков, как в предыдущем примере, не всегда оправданно. Для хранения и обработки потоков в Java существует класс ThreadGroup. Группа, к которой принадлежит создаваемый поток, опять-таки передается конструктору Thread():

```
ThreadGroup tg = new ThreadGroup("NameThre
adGroup");
Thread thread = new Thread(tg, new
SecondThread(), "ThreadName");
```

Если группа не указана явно, поток будет помещен в тот же ThreadGroup, что и его родитель.

```
  ThreadGroup tg = new ThreadGroup("NameThreadGroup");
  Thread thread = new Thread(tg, new SecondThread(), "ThreadName");
  Thread thread1 = new Thread(tg, new SecondThread(),
"ThreadName2");
  Thread thread2 = new Thread(tg, new SecondThread(),
"ThreadName3");
  System.out.println("active thread in group " + tg.activeCount());
  thread.start();
  thread1.start();
  System.out.println("active thread in group " + tg.activeCount());
  Thread[] threads = new Thread[tg.activeCount()];
  int m = tg.enumerate(threads);
  System.out.println("taked threads from group : " + m);
  for (int i = 0; i < threads.length; i++) {
   System.out.println(threads[i].getName());
  }
```

В представленном выше примере в экземпляр класса ThreadGroup помещаются три потока, два из которых запускаются на выполнение. Количество активных потоков в группе определяется с помощью метода activeCount(), а в результате выполнения метода enumerate() формируется перечень всех активных потоков.

Управление потоками

При запуске потоков следует учитывать и то, что их иногда приходится останавливать, причем как штатно, так и экстренно. Для того, чтобы приостановить работу потока изнутри, допустим, в тот момент, когда закончилась доступные для обработки данные, можно использовать два метода: sleep() и wait().

```
public class ThirdThread extends Thread {
 public void run() {
  for (int i = 1; i < 110; i++) {
   if (i == 10) {
    try { sleep(10000);
    } catch (InterruptedException e) {
                    System.out.println("the thread was awaken there
(just a moment ago)");
    }}
   if (i == 100) {
    try {
     synchronized (this) { wait();}
    } catch (InterruptedException e) {
                    System.out.println("the thread was awaken there
(just a moment ago)again");
    }
   }
  }}}
```

Методу sleep() передается переменная типа long, соответствующая количеству миллисекунд, в течении которых поток будет «спать». В случае wait() поток ждет пробуждения снаружи. Применение методов sleep() и wait() требует обработки исключительной ситуации, которая возникают при пробуждении потока. В представленном ниже классе ConsoleToThreadThree потоки пробуждаются с помощью метода interrupt():

```
public class ConsoleToThreadThree {
 public static void main(String[] args) {
  ThirdThread thread = new ThirdThread();
  thread.start();
  for (int i = 1; i < 20; i++) {
   System.out.println("It is in main " + i);
  }
  thread.interrupt();
  for (int i = 1; i < 20; i++) {
   System.out.println("It is in main " + i);
  }
  thread.interrupt();
  try {
   thread.join();
  } catch (InterruptedException ex) {
   System.out.println("Exception in stop thread");
  }}}
```

Отметим, что вызов метода wait() без блока синхронизации (synchronized), о котором мы поговорим чуть ниже, приводит к исключению IllegalMonitorStateException. Возникшая ошибка свидетельствует об отсутствии монитора у объекта (понятие монитора и синхронизация тесно связаны, о чем мы тоже поговорим ниже). Если для приостановления потока был применен метод wait(), то для «пробуждения» потока можно воспользоваться методом notify() или notifyAll(). Первый пробуждает один случайно выбранный спящий поток, а второй пытается пробудить их всех.

Кроме рассмотренных выше методов, иногда бывает целесообразно использовать метод yield(), который приостанавливает работу текущего потока. Метод yield() не переводит поток в режим ожидания, как wait(), но предоставляет другим потокам возможность начать работать раньше, чем допустила бы Java-машина [фактически, поток, вызвавший yield() добровольно отдает свой квант процессорного времени, – прим. ред.]. Метод yield() статичный, так что прекратить с его помощью работу другого потока не получится.

Методов остановки потоков тоже нет (ранее присутствовали методы stop(), resume(), suspend(), но сейчас они объявлены как «deprecated» – то есть нерекомендованными к использованию). На сегодня в Java принят уведомительный стиль остановки потока с помощью пары методов: уже известного нам interrupt(), применяемого снаружи, чтобы выставить флаг завершения и метода isInterrupted(), вызываемого изнутри потока, чтобы узнать состояние флага, свидетельствующего о том, что «пора закругляться».

Мониторы и синхронизация

Что такое «монитор», о котором говорилось выше? Нет, это не дисплей, это – объект, используемый как защелка, то есть в данный момент времени владеть монитором может только один поток. В случае, если поток завладел монитором, говорят, что он «вошел» в монитор, а все остальные потоки, пытающиеся это сделать, будут заморожены (часто говорят, что они «ждут» монитора) до тех пор, пока владелец монитора его не освободит, то есть не покинет.

Если перейти к реалиям Java, то объектов типа монитор в явном виде просто нет! С каждым объектом связан неявный монитор, и чтобы завладеть им, необходимо вызвать метод или блок, помеченный ключевым словом synchronized. Как только поток входит в такой блок, он завладевает монитором объекта, переданного synchronized в качестве параметра. Так происходит и в классе ThirdThread, однако, в момент вызова метода wait(), монитор отпускается. Пример synchronized-метода представлен ниже:

```
 public synchronized boolean sameCheck() {
  if (a) {
   a = false; return true;
  } else {
```

```
    a = true; return false;
  }}
```

В целом, синхронизация – это механизм, обеспечивающий монопольный доступ участка кода к некоторому объекту. Одним из первых способов, предложенных для синхронизации работы потоков, были семафоры, концепцию которых описал Дейкстра [Dijkstra] в 1965 году (часто говорят, что семафор – это классический синхронизирован-ный примитив). Семафор используется для предоставления доступа к ограниченному количеству ресурсов. Как правило, у семафора есть две операции: P – занять ресурс и V – освободить ресурс. Он может быть реализован следующим образом:

```
public class SimpleSemaphore {
  int counter;
  public SimpleSemaphore() {
    this.counter = 1;
  }

  public synchronized void p() throws InterruptedException {
    while (counter == 0) {
      wait();
    }
    counter = counter + 1;
  }
  public synchronized void v() throws InterruptedException {
    counter = counter - 1;
    notify();
  }
}
```

Можно, конечно, реализовывать семафоры самостоятельно, но проще воспользоваться специальной библиотекой java.util.concurrent. Кроме семафоров, она включает в себя еще много чего интересного.

Отметим, что программа, в принципе, может не использовать ни один из методов синхронизации, обходясь методами wait() и notify().

Взаимные блокировки

На этом наш рассказ можно было бы и завершить, но чтобы у вас не сложилось впечатление, что в мире многопоточных приложений все так радужно, мы поговорим о неприятных последствиях синхронизации. Как только количество потоков начинает стремительно расти и возникает необходимость синхронизированного доступа к ограниченному кругу объектов в различной последовательности, будьте готовы к ошибкам типа deadlock – взаимным блокировкам.

Взаимная блокировка – это ошибка, которая лучше всего описывается простой формулой: «Поток A держит монитор a и хочет захватить монитор b, а поток B держит монитор b и хочет захватить монитор a». В результате оба засыпают «мертвым сном».

Ошибка очень противная и возникает обычно в нетривиальных алгоритмах. Лечится взаимная блокировка грамотным проектированием и профилактическими мерами, вроде следующей: всегда захватывайте мониторы в одном и том же порядке.

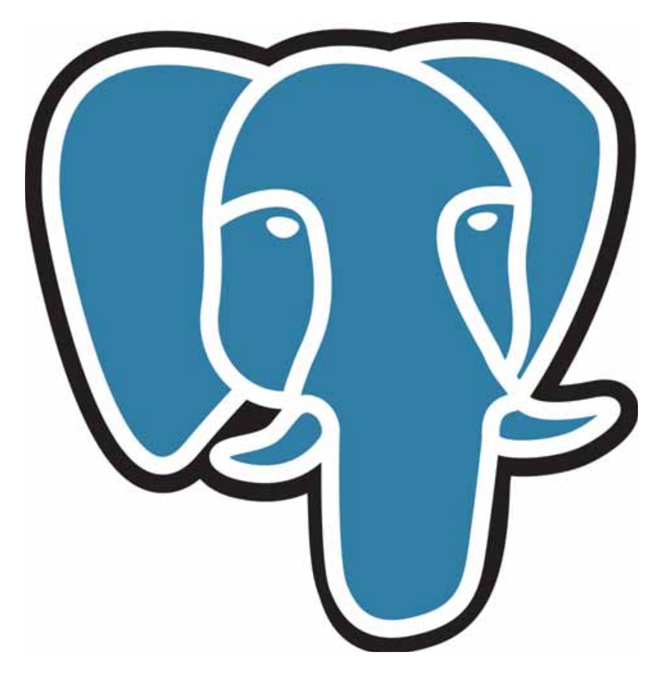
***

Сегодня мы поговорили о двух способах создания потоков Java, разобрались с приоритетами, познакомились со средствами управления работой потоков и демонами, а также сделали небольшой обзор методов синхронизации. Для того, чтобы начать практическую работу с потоками, этого вполне достаточно. Желающим разобраться во всем этом глубже я рекомендую ознакомится с книгой «Concurrent Programming in Java: Design Principles and Patterns», автором которой является Дуг Ли [Doug Lea] – она считается одной из лучших по данной тематике.

На этом мы заканчиваем обзор основ программирования на Java и в следующий раз поговорим о серверных приложениях – приготовьтесь к Java Enterprise Edition! LXF

Работа

ЧАСТЬ 3: Вы уже неоднократно слышали, что PostgreSQL – это одна из самых мощных открытых СУБД, и сегодня **Евгений Балдин** расскажет вам, почему.



«Познание бесконечности требует бесконечного времени».
Девиз отдела Абсолютного Знания
«Понедельник начинается в субботу»

В этой статье мы предпримем попытку сделать краткий обзор возможностей *PostgreSQL*. Не надо иллюзий – «объять необъятное невозможно», поэтому многое из интересного осталось за кадром, но все же «попытка – не пытка».

Немного об основах

Применительно к базам данных часто упоминается принцип ACID: атомарность (Atomicity), целостность (Consistency), локализация пользовательских процессов (Isolation) и устойчивость к ошибкам (Durability).

Для обеспечения совместной работы множества пользователей (concurrency) в целях следования заветам ACID, *PostgreSQL* использует систему управления версиями или MVCC (Multi-Version Concurrency Control). При подсоединении пользователя MVCC «подсовывает» ему собственную версию или мгновенный снимок (snapshot) базы данных. В этом случае изменения, производимые пользователем, невидимы другими пользователями до тех пор, пока текущая транзакция[1] (transaction) не подтверждается (commit). Кроме проблем, связанных с ACID, многоверсионность позволяет уменьшить или даже, во многих случаях, исключить необходимость блокировки данных (locks) при чтении.

Надежность[2] (reliability) сохранения данных является одним из основных показателей качества СУБД. Сохранение измененных данных – очень нетривиальная процедура. Дело в том, что диски очень «ме-е-едленные», поэтому прежде чем попасть на диск, данные проходят через промежуточные буферы (cache), начиная от системного кэша файловой системы, заканчивая кэшем на самом диске. Никто не сможет гарантировать, что все в них положенное, в случае возникновения каких-либо проблем, окажется в безопасном постоянном хранилище. Для максимального уменьшения вероятности потери данных *PostgreSQL* использует журнал транзакций или Write Ahead Log (WAL). Прежде чем записать данные о проведенной транзакции на диск, информация об изменениях заносится в WAL. Если что-то случилось, то данные можно восстановить по журналу. Если данные не попали в журнал, то, соответственно, исчезнет вся транзакция – жалко, конечно, зато целостность не нарушается. Следствием использования WAL является отсутствие необходимости «скидывать» данные на диск с помощью fsync, так как достаточно убедиться, что записан WAL. Это значительно увеличивает производительность в многопользовательской среде с множеством мелких запросов на изменение данных, так как записать один последовательный файл WAL гораздо проще, чем изменять множество таблиц по всему диску. В качестве бонуса журнал транзакций позволяет организовать непрерывное резервное копирование данных (on-line backup) – мечту администратора и возможность «отката» базы данных на любой момент в прошлом (point-in-time recovery) – своеобразную машину времени.

Типы данных

Как и положено базе данных, *PostgreSQL* поддерживает довольно много стандартных типов данных. Более того, пользователь может определять свои собственные типы данных, если он не найдет необходимых примитивов среди стандартных.

Числовые типы

Обычные числовые (numeric) типы представлены целыми числами в два (smallint), четыре (integer) или восемь (bigint) байт длиной, числа с плавающей точкой – в четыре (real) и восемь байт (double precision) длиной. Кроме обычных чисел, в случае real и double поддерживаются значения Infinity, -Infinity и NaN – бесконечность (∞), минус бесконечность (-∞) и «не число» (not-a-number), соответственно.

[1] Транзакция представляет собой последовательность операций, которая обязана либо выполниться полностью, либо отмениться совсем, как будто это единое целое. При этом, независимо от других параллельно идущих транзакций (isolation), должна сохраняться целостность данных (consistency).

[2] Этот момент отражен в FAQ **fido7.ru.os.cmp** следующим образом:
Q51: Народ, а вы стабильным софтом пользоваться не пробовали?
A51: Пробовали, но мэйнфреймы с дизель-генераторами не везде есть.

Месяц назад Мы исследовали различных клиентов для удобной работы с PostgreSQL.

с базой

PostgreSQL поддерживает числа с произвольной точностью numeric(precision,scale), где precision – число всех знаков в определяемой величине, а scale – число знаков в дробной части. *PostgreSQL* позволяет выполнять действия без накопления ошибки с подобными величинами с точностью вплоть до 1000 знаков. Этим типом данных не следует злоупотреблять, так как операции над подобными числами занимают очень много времени.

Битовые поля представлены типами bit(size) – битовая строка фиксированной длины size и bit varying(size) – битовая строка переменной длины с ограничением по размеру size.

К числовым типам *PostgreSQL* относятся и «псевдотипы» serial и bigserial. Эти типы соответствуют типам integer и bigint, за исключением того, что при записи новых данных в таблицу с колонкой этого типа, значение по умолчанию в ней увеличивается на единицу – то есть это автоматически создаваемая упорядоченная последовательность.

Символьные типы

В стандарте *SQL* символьный тип определяется как строка определенной длины character(size), где size – длина строки. В дополнение к стандарту, *PostgreSQL* поддерживает строки переменной длины с ограничением varchar(size) и без ограничения – text.

Бинарные типы

Бинарную строку можно сохранить, используя тип bytea. *SQL* предполагает, что вся информация передается как текст, поэтому при передаче данных следует экранировать некоторые из символов.

В *PostgreSQL* есть специальный тип данных Large Objects. По сути дела, это просто способ сохранять любые файлы размером вплоть до 2 Гб прямо в базе данных. Операции с подобными объектами выходит за рамки *SQL*. Для доступа к Large Objects есть специальный программный интерфейс, смоделированный по образу и подобию обычного чтения/записи файла.

Типы даты/времени

Временем в *PostgreSQL* заведует тип timestamp или timestamp with time zone – он может хранить дату и время, начиная с 4713 г. до н.э. вплоть до 5874897 г., с точностью в одну микросекунду (µс), занимая восемь байт. Второй упомянутый тип включает часовой пояс и позволяет автоматически учитывать переход на летнее/зимнее время. С таким диапазоном и точностью проблема типа широко разрекламированной Y2K случится не скоро.

Разница между двумя датами представлена типом interval длиной в двенадцать байт, что позволяет хранить информацию о событиях, связанных даже с рождением и смертью Вселенной.

Также есть отдельный тип для календарного времени (date) и просто для времени (time или time with timezone).

PostgreSQL поддерживает множество способов ввода даты и времени. С моей точки зрения, в некоторых случаях СУБД проявляет излишний интеллект, поэтому рекомендую выбрать стандартный ISO, который выглядит примерно так:

```
test=> -- узнаем текущее время с точностью до секунды
test=> select date_trunc('seconds',timestamp with time zone 'now');
  date_trunc
------------------------
 2006-08-26 21:08:14+07
```

В таком случае ошибиться в порядке месяца и дня становится весьма затруднительно, независимо от того, какая локаль используется системой.

Для типа timestamp определены дополнительные константы:

- epoch – начало эпохи с точки зрения времени Unix (четырехбайтовый time_t): **1970-01-01 00:00:00+00;**
- infinity – позже, чем любое возможное из времен;
- -infinity – раньше, чем любое возможное из времен;
- now – здесь и сейчас;
- today – сегодняшняя полночь; аналогично, yesterday – вчерашняя полночь и, tomorrow – завтрашняя полночь.

Логические типы

Логические типы представлены типом boolean. Логично, что он принимает значения либо **TRUE** ('**t**', '**true**', '**y**','**yes**', '**1**') – «истина», либо **FALSE** ('**f**', '**false**', '**n**', '**no**', '**0**') – «ложь». Все просто, за исключением одного «но» – есть еще одна возможность: «значение не определено» (**NULL**). Собственно говоря, это не особенность типа boolean. При использовании *SQL*, с тем, что значение может быть не определено, необходимо считаться всегда и везде. Вот такая вот логика – вовсе не двоичная.

Остальные стандартные типы

К оставшимся стандартным типа относятся различные геометрические типы данных: точка (point), линия (line), отрезок (lseg), прямоугольник (box), путь (path), замкнутый путь (polygon) и окружность (circle). Для системных администраторов будут интересны стандартные типы сетевых IPv4 и IPv6-адресов (cidr или inet) и тип MAC-адреса (macaddr).

Более сложные типы реализуются как дополнения. Яркими примерами служат поддержка географических объектов GIS (**http://postgis.refractions.net/**) и иерархический тип данных ltree (contrib/ltree).

Определение пользовательских типов

Прежде всего следует упомянуть, что *PostgreSQL* поддерживает массивы. Можно создать массив определенного размера или безразмерный, на основе любого стандартного типа или типа, определенного пользователем. Поддерживаются многомерные массивы и операции над ними, например, «срезы».

```
test=> --- создаем массив для игры в «крестики-нолики»
test=> create table tictactoe (squares integer[3][3]);
test=> --- |x00| x = 1, 0 = -1
test=> --- |0xx| вставляем информацию о варианте игры
test=> --- | x| крестики начинают и выигрывают
test=> insert into tictactoe
test->   values ('{{1,-1,-1},{-1,1,1},{0,0,1}}');
test=> --- распечатываем сохраненную позицию
test=> select * from tictactoe ;
    squares
------------------------------
 {{1,-1,-1},{-1,1,1},{0,0,1}}
test=> -- распечатываем значение первого столбца
test=> select squares[1:3][1:1] from tictactoe ;
  squares
---------------
 {{1},{-1},{0}}
```

Композитный тип (composite type) представляет из себя аналог структуры:

```
test=> CREATE TYPE complex AS (Re real,Im real);
```

В отличие от стандартных встроенных типов, использование композитного типа пока имеет некоторые ограничения. Например, из них нельзя создавать массивы. »

» *PostgreSQL* позволяет выйти за рамки стандартного *SQL* в целях создания пользовательских типов данных и операций над ними. Подробнее об этом можно узнать, изучив документацию по команде CREATE TYPE.

Функции

Все стандартные типы имеют свои функции: ведь если есть тип, то с ним нужно работать. Стандартные функций многочисленны[3] и разнообразны. Одних операторов поиска с использованием регулярных выражений – целых три штуки: собственное расширение *PostgreSQL* (LIKE и ILIKE), оператор, соответствующий стандарту *SQL* (SIMILAR TO) и POSIX-совместимый оператор (~ и ~*). Все, что только можно было быстро придумать, уже реализовано. А более сложные случаи, например, модуль для полнотекстового поиска tsearch2 (contrib/tsearch2) находятся в процессе совершенствования. Изобрести что-то, выходящее за рамки стандарта тяжело, но если это случится, вы всегда можете создать свою собственную функцию. При желании, ссылаясь на уже имеющуюся функцию, с помощью команды CREATE OPERATOR можно определить оператор для своих типов данных.

Хранимые процедуры

Для создания новых функций используется оператор CREATE FUNCTION, что вполне предсказуемо. Создаваемые таким образом функции исполняются и хранятся на сервере, отсюда и название – «хранимые процедуры»:

```
test=> -- Создаем и заполняем таблицу
test=> create table AplusB (A integer, B integer);
test=> insert INTO AplusB VALUES (1,1);
test=> insert INTO AplusB VALUES (2,2);
test=> insert INTO AplusB VALUES (3,3);
test=> -- Создаем новую функцию
test=> CREATE FUNCTION plus(integer, integer) RETURNS integer
test->   LANGUAGE SQL as 'SELECT $1 + $2;';
CREATE FUNCTION
test=> select A,B,plus(A,B) from AplusB;
 a | b | plus
-----+-----+------
  1 |  1 |  2
  2 |  2 |  4
  3 |  3 |  6
(записей: 3)
```

PostgreSQL поддерживает перегрузку функций: объектно-ориентированность имеет свои плюсы. Кроме *SQL*, для создания новых функций можно использовать процедурные языки программирования. Для начала работы с процедурным языком его необходимо инициализировать. По умолчанию, интерфейсы к языкам, отличным от *SQL* и C, недоступны по соображениям безопасности. Для инициализации языка используется команда createlang. Запустить ее может только администратор базы данных – тот, кто имеет право создавать базы:

```
# Инициализируем язык PL/pgSQL для базы данных test
> createlang plpgsql test
# делаем то же самое, но для языка PL/Perl
> createlang plperl test
```

Теперь можно создавать функции с использованием всех прелестей процедурного программирования, вместе с циклами, каковые по понятным причинам отсутствуют в *SQL*. Ниже продублирована простейшая функция, которая была описана выше, но теперь уже на PL/pgSQL и на PL/Perl:

```
test=> -- Создаем новую функцию с использование PL/pgSQL
test=> CREATE FUNCTION pgsql_plus(integer, integer) RETURNS integer
test->   LANGUAGE PLPGSQL as 'BEGIN return $1+$2; END;';
CREATE FUNCTION
test=> -- Создаем новую функцию с использование PL/Perl
test=> CREATE FUNCTION perl_plus(integer, integer) RETURNS integer
test->   LANGUAGE PLPERL as 'return $_[0]+$_[1]';
CREATE FUNCTION
test=> -- Проверяем, что все работает
test=> select pgsql_plus(A,B) from AplusB;
test=> select plus(A,B),pgsql_plus(A,B),perl_plus(A,B) from AplusB;
 plus | pgsql_plus | perl_plus
------+------------+-----------
   2 |     2 |     2
   4 |     4 |     4
   6 |     6 |     6
(записей: 3)
```

В стандартной документации подробно описаны идущие вместе с дистрибутивом языки: PL/pgSQL, PL/Tcl, PL/Perl, PL/Python и, естественно, C/C++ с *SQL*. Кроме перечисленных здесь, есть поддержка

- **plPHP** http://plphp.commandprompt.com/,
- **PL/java** http://gborg.PostgreSQL.org/project/pljava/projdisplay.php,
- **PL/R** http://www.joeconway.com/plr/,
- **PL/Ruby** http://raa.ruby-lang.org/project/pl-ruby,
- **PL/sh** http://plsh.projects.PostgreSQL.org/.

Также есть возможность подключения вашего любимого языка.

Триггеры

Обычно для решения несложных задач можно удовлетвориться сценарием: «что сказано – то и сделано», но в более сложных случаях от СУБД хотелось бы получать более сложные реакции в ответ на внешнее «раздражение». Для управления реакцией СУБД на изменение данных используются триггеры. Для создания триггера используется команда CREATE TRIGGER.

Полное описание команды в форме Бэкуса-Наура таково:

```
CREATE TRIGGER «имя триггера»
 { BEFORE | AFTER } { «событие» [ OR ... ] }
 ON «имя таблицы» [ FOR [ EACH ] { ROW | STATEMENT } ]
 EXECUTE PROCEDURE «исполняемая функция - реакция»
```

Реакция на «событие», которое может быть вставкой (INSERT), изменением (UPDATE), или удалением (DELETE) может производиться, по выбору, до (BEFORE) или после (AFTER) изменения данных. Выполнение процедуры может производиться для каждой записи (ROW) или для каждого запроса (STATEMENT). Для показательного примера создания триггера возьмем следующую выдуманную задачу: при изменении данных в описанной уже таблице AplusB сумма A и B должна автоматически обновляться в таблице ABresult. Следующее решение чрезвычайно неоптимально, зато работает:

```
test=> -- Создаем «результирующую» таблицу
test=> create table ABresult (result integer);
test=> -- Создаем функцию, очищающую ABresult и
test=> -- заполняющую все суммой A и B из AplusB.
test=> create function ABsumm() returns trigger as
test-> 'BEGIN
test'>     delete from ABresult;
test'>     insert into ABresult values (AplusB.A+AplusB.B);
test'>     return NULL;
test'> END;'
test-> language 'plpgsql';
test=> -- Создаем триггер
test=> CREATE TRIGGER makeABresult
test=>     AFTER INSERT or UPDATE or DELETE on AplusB
test=>     FOR EACH STATEMENT execute procedure ABsumm();
CREATE TRIGGER
test=> -- Добавляем данных в таблицу AplusB
test=> insert into AplusB VALUES (100,200);
test=> -- проверяем, что триггер сработал
test=> select * from AplusB,ABresult where A+B=result;
 a | b | result
-----+-----+--------
```

[3] Их больше 1500. Полный список можно вывести, набрав в *psql* команду \df.

```
  1 |  1 |  2
  2 |  2 |  4
  3 |  3 |  6
100 | 200 | 300
(записей: 4)
```

Rules

Кроме триггеров, *PostgreSQL* обладает еще одним способом управления реакции СУБД на запросы – это rules, или «правила». Для создания «правил» используется команда CREATE RULE. Основное отличие «правила» от триггера в том, что триггер – это реакция системы на изменение данных, а «правило» позволяет изменять сам запрос, в том числе и запрос на получение данных (SELECT). В частности, одно из довольно удобных расширений *PostgreSQL* – представление или виртуальная таблица (view), реализовано с помощью «правил».

Индексы

Традиционно, для ускорения поиска информацию индексируют. Если данных немного, то можно прожить и так. Серьезные же задачи требуют серьезных объемов, поэтому без индексов не обойтись.

Создание индексов – это ответственность создателя БД. Создание индекса, как можно догадаться, производится с помощью команды CREATE INDEX:

```
CREATE [ UNIQUE ] INDEX «имя индекса» ON table [ USING «алгоритм» ]
  ( { «имя столбца» | ( «выражение» ) } [, ...] )
  [ WHERE «условие» ]
```

Индекс может быть уникальным (UNIQUE). В этом случае при создании индекса и при добавлении данных накладывается дополнительное требование на уникальность параметра, по которому создается индекс.

При создании индекса можно выбрать алгоритм индексации. По умолчанию используется B-tree, но доступны также Hash, R-tree или GiST. Алгоритм GiST (http://www.sai.msu.su/~megera/postgres/gist/) был создан Олегом Бартуновым на пару с Федором Сигаевым. GiST является не просто еще одним алгоритмом – это целый конструктор, позволяющим создавать индексы для принципиально новых типов данных. В *PostgreSQL 8.2* будут добавлены еще два метода: bitmap и GIN. Если судить по алгоритмам создания индексов, то *PostgreSQL* – это одна из самых продвинутых СУБД.

Индекс можно создавать по какому-то из столбцов – это самый простой метод. При указании нескольких колонок создаются многоколоночные индексы. Особо следует отметить возможность создания функциональных индексов – в качестве индекса указывается функция от данных таблицы. С помощью функциональных индексов можно реализовать еще один алгоритм индексации: Reverse index (обращает поле переменной – первый символ считается последним).

Условие (WHERE), накладываемое при создании индекса, позволяет создавать частичные индексы (partial indices). Это полезно в тех случаях, когда индексируемый столбец содержит большое число одинаковых значений, а поиск надо производить по редким «чужеродным» вкраплениям.

Для того, чтобы индекс работал, как надо, необходимо следить, чтобы в базе данных регулярно запускалась процедура ANALYZE, которая собирает статистику о распределении значений в индексах. Собранная статистика, в свою очередь, позволяет планировщику верно принимать решение о порядке выполнения запроса. Для оптимизации поиска информации временами может оказаться полезна собственная команда *PostgreSQL* CLUSTER. С ее помощью можно упорядочить записи в таблице согласно указанному индексу.

Целостность данных

Сохранить, записать, а затем быстро достать данные – вещь полезная, но как отследить, что они записаны правильно и без ошибок? Для это-

PostgreSQL в лицах: Алексей Борзов



Визитка LXF:
Окончил ВМиК МГУ в 2000 году, с тех пор занимается web-программированием. Место работы: «свободный художник». Домашняя страничка отсутствует – по всей видимости это уже давно пройденный этап.

Евгений М. Балдин (ЕМБ): Как вы начали использовать PostgreSQL?

Алексей В. Борзов (АВБ): С *PostgreSQL* я работаю с 2000 года. Тогда меня взяли в Издательский дом «РДВ Медиа» переделывать «слепленный на коленке» сайт газеты «Работа для вас» (ныне http://rabota.ru/). Новый сайт с самого начала разрабатывался на связке *PHP+PostgreSQL*. Надо признать, *PostgreSQL* не был моим выбором – мне его порекомендовали.

ЕМБ: На сайте PostgreSQL написано, что вы «wrote the majority of the main website's framework». Что сподвигло на такой подвиг?

АВБ: Причина того, что я «впрягся», банальна – на старую версию сайта *PostgreSQL* нельзя было смотреть без слез, и тенденции к улучшению ситуации не наблюдалось.

На момент «прикладывания рук» к сайту *PostgreSQL* у меня был уже опыт разработки Open Source. Я поддерживаю/пишу несколько пакетов в репозитории PEAR (http://pear.php.net/user/avb), посему для меня вполне естественно было предложить свои услуги.

ЕМБ: Насколько это было сложно? Довольны ли результатом?

АВБ: С технической точки зрения, в написании сайта для *PostgreSQL* была только одна трудность: сайт должен поддерживать возможность быть разнесенным по зеркалам, а требовать от владельцев зеркал, чтобы они поднимали у себя *PostgreSQL* и реплицировали данные с центрального сервера, практически нереально. Поэтому сайт писался сразу так, чтобы все динамические действия выполнялись на центральном сервере, а зеркала получали только статический HTML. А так – сайт далеко не самый сложный из тех, которые мне приходилось делать.

С организационной же точки зрения, группа поддержки web-сайтов *PostgreSQL* отличается немалым разгильдяйством. Информацию о том, что и как надо сделать, приходилось «выбивать».

В целом, получилось неплохо. Нашлись профессиональные дизайнеры, и ключевая в истории проекта *PostgreSQL* версия 8.0 анонсировалась уже на респектабельно выглядящем сайте, а не на убогой домашней страничке.

ЕМБ: Вы активно используете в своей работе связку *PHP+PostgreSQL*. На сколько это связка естественна? Что мешает *PostgreSQL* потеснить M (*MySQL*) из *LAMP*[5]?

АВБ: Язык PHP позволяет работать с огромным количеством различных СУБД, причем, как правило, используя их «родные» интерфейсы. Поэтому связка *PHP+PostgreSQL* вполне естественна, так же как и связка *PHP+Oracle* и т.д. Тот факт, что PHP упоминается обычно в виде *PHP+MySQL* или в аббревиатуре LAMP – исключительно результат маркетинга. На тему помех «вытеснению буквы M» у меня есть пара очевидных соображений:

» Версия *PostgreSQL* для Windows вышла не так давно, а большая часть разработчиков все же сидят под Windows. Я понимаю, не очень хорошая фраза для журнала *Linux Format*, но…

» Опять же, большая часть разработчиков имеет очень плохую подготовку, поэтому легко «ведется» на рассказы о том, что одна СУБД может всегда работать в 10 раз быстрее другой, что транзакции и внешние ключи придумали трусы и т.д., и т.п. К тому же, для этих товарищей выучить даже что-то одно – практически непосильный труд, поэтому речь об изучении другой технологии уже просто не идет.

[5] Один из вариантов расшифровки LAMP=Linux+Apache+MySQL+PHP.

го необходимо постоянно следить за целостностью данных в условиях многопользовательской системы.

Транзакции

Транзакция – это единый блок операций, который нельзя разорвать. Блок либо выполняется целиком, либо все отменяется. В условиях параллельного доступа, *PostgreSQL* распространяет информацию об операциях только по завершению транзакции. Транзакция начинается с оператора BEGIN и заканчивается оператором COMMIT (подтверждение транзакции) или ROLLBACK (отмена транзакции). Возможен режим, когда каждый запрос сам себе является транзакцией: например, такой режим по умолчанию используется в *pSQL*. Для отмены этого режима достаточно набрать BEGIN. Неудобством при использовании транзакций является то, что в случае ошибки какого-то из запросов приходится отменять всю транзакцию. Для устранения этого недостатка в *PostgreSQL 8.x* были добавлены точки сохранения (savepoints).

```
test=> -- начинаем транзакцию
test=> BEGIN;
test=> -- здесь идет блок операторов, который удачно завершается

test=> -- ставим метку
test=> SAVEPOINT savepoint_one;
test=> -- здесь идет блок операторов, в котором произошла ошибка

test=> -- откатываемся до установленной метки,
test=> -- а не отменяем всю транзакцию
test=> ROLLBACK TO savepoint_one;
test=> -- повторяем последний блок

test=> -- завершаем транзакцию
test=> COMMIT;
test=> -- все, теперь изменения доступны всем
```

Ограничения

Целостность данных обеспечивает не только многоверсионностью (MVCC) *PostgreSQL*, но и «архитектором» таблиц базы данных. При создании таблицы (CREATE TABLE) или позже можно добавить ограничение (CONSTRAINT) на диапазон записываемых в таблицу данных. Ограничением могут быть как простые арифметические условные выражения, требования уникальности (UNIQUE или PRIMARY KEY), так и более сложные ограничения в виде внешних ключей (FOREIGN KEY).

Если какой-то столбец A является внешним ключом (FOREIGN KEY) по отношению к столбцу B (REFERENCES), то это означает, что только данные, представленные в столбце B, могут появиться в качестве значений столбца A. В случае внешних ключей *PostgreSQL* осуществляет автоматический контроль ссылочной целостности[4]. Это довольно интересный механизм, который, в частности, позволяет моделировать иерархические структуры.

Блокировки

Поскольку в условиях параллельного доступа к базе данных каждый пользователь работает со своим мгновенным снимком (следствие MVCC), а не с самой БД, то в принципе можно придумать ситуацию, когда полученные данные «устаревают», так как они были изменены другим пользователем. Если это обстоятельство важно, то *PostgreSQL* предоставляет полный ассортимент блокировок. С помощью команды LOCK можно заблокировать таблицу, а инструкция SELECT FOR UPDATE позволяет заблокировать отдельные записи. Следует учитывать, что использование блокировок увеличивает шанс взаимной блокировки (deadlock). *PostgreSQL* умеет определять взаимные блокировки и разрешать их путем прекращения одной из транзакций, но на это уходит время.

[4] Ссылочная целостность — гарантированное отсутствие внешних ключей, ссылающихся на несуществующие записи в этой или других таблицах.

Азбука SQL: B

Select

Самый популярный оператор в *SQL* – это SELECT. Получение данных практически всегда организуется с его помощью. Он всего один, поэтому нет никакого языка получения данных – он сам себе язык. Причем язык декларативный, потому как при использовании SELECT описываются свойства искомых данных, а не информация о том, как эти данные получить.

Урезанные правила для SELECT представлены ниже:

```
SELECT «список искомых данных»
  [ FROM «список источников получения данных» ]
  [ WHERE «условное выражение» ]
  [ GROUP BY «выражение» [, ...]]
  [ HAVING «условное выражение» [, ...] ]
  [ ORDER BY «выражение» [, ...] ]
  [ LIMIT «число» ]
  [ OFFSET «число» ]
```

Если в качестве «списка искомых данных» передать «*» (звездочку), то выводятся все поля из «списка источников получения данных». Результат, выдаваемый SELECT, можно использовать как источник получения данных, наравне с именами таблиц. Вложенный SELECT должен заключаться в круглые скобки. С помощью WHERE описываются свойства, которые хочется видеть среди полученных данных.

Инструкция GROUP BY позволяет сгруппировать результаты по указанному признаку. Инструкция HAVING выполняет примерно те же функции, что и инструкция WHERE, но работает уже после применения GROUP BY. Поэтому в случае HAVING можно использовать агрегатные функции.

Сортировка данных обеспечивается с помощью инструкции ORDER BY. «Выражение» для сортировки представляет из себя функцию от имен столбцов (просто имя столбца – тоже выражение) и метода сортировки. Через запятую можно указать еще одно выражение, которое принимается во внимание, если предыдущее выражение для сортируемых строчек выдает одинаковые значения. Сортировка может производиться по возрастанию (ASC), по убыванию (DESC) и по заданному пользователем алгоритму (USING «оператор сравнения»; ASC эквивалентен конструкции USING <, DESC – инструкции USING >.).

Для ограничения на число получаемых в ответ на запрос строк можно использовать инструкцию LIMIT, которая гарантирует, что число выведенных строк не будет превышать указанное в качестве параметра число. Инструкция OFFSET указывает SELECT, сколько строк из уже отобранных следует пропустить, прежде чем начать вывод.

Послесловие

Хотелось бы еще раз сказать, что «нельзя объять необъятное». Единственная проблема состоит в том, что конкретно это рассматриваемое нами «необъятное» уже «объято», поэтому за всеми подробностями следует обратиться к стандартной документации, а в качестве бонуса рекомендую хорошую обзорную статью от Олега Бартунова: «Что такое *PostgreSQL*?»: http://www.sai.msu.su/~Emegera/postgres/talks/what_is_PostgreSQL.html. LXF

Через месяц Мы займемся программными интерфейсами *PostgreSQL*, в первую очередь – *libpq*.



DESTEN®
COMPUTERS

Документация и

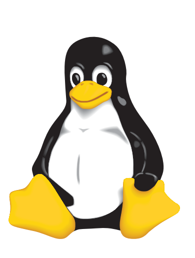
ЧАСТЬ 4: TeX, как известно, был создан для представления кода и алгоритмов. Так постигайте же замысел создателя и его последователей вместе с **Евгением Балдиным**!

+++ Ошибка Деления На Огурец.
Переустановите Вселенную И Перезагрузитесь +++
Так зависает Гекс.

Источник: «Санта-Хрякус» от Терри Пратчетта









Программирование под Linux – вполне естественное занятие. Написание документации – неотъемлемая часть этого процесса. *LaTeX* достоин быть включённым в технологическую цепочку по выпуску программного продукта.

Если вспомнить историю, то Д.Э. Кнут создал TeX именно для целей представления кода и алгоритмов в своём глобальном пятитомнике «Искусство программирования».



Спецсредства

Чтобы украсить инструкцию, надо добавлять в неё «пятна». Злоупотреблять этим не стоит, но пару мыслей выделить вполне реально. Упомянутые ниже приёмы далеко – не все, что может предложить *LaTeX*: это просто демонстрация возможностей.

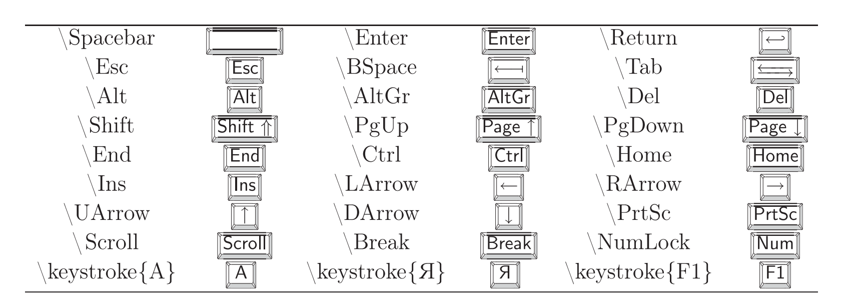
keystroke

Иногда в тексте необходимы фразы вида: «Для выхода из программы нужно нажать клавишу Esc.» Макрос \keystroke, определённый в одноимённом пакете *keystroke*, позволяет выделить название клавиши, примерно следующим образом:

```
Для продолжения нажмите
\keystroke{<<любую клавишу>>}.
```



В пакете определены многие клавиши, имеющиеся на стандартной клавиатуре. Пакет очень прост и имеет зачатки интернационализации – его легко адаптировать под свои нужды.



\Spacebar		\Enter	Enter	\Return	↵
\Esc	Esc	\BSpace	⟵	\Tab	⇆
\Alt	Alt	\AltGr	AltGr	\Del	Del
\Shift	Shift ⇑	\PgUp	Page ↑	\PgDown	Page ↓
\End	End	\Ctrl	Ctrl	\Home	Home
\Ins	Ins	\LArrow	←	\RArrow	→
\UArrow	↑	\DArrow	↓	\PrtSc	PrtSc
\Scroll	Scroll	\Break	Break	\NumLock	Num
\keystroke{A}	A	\keystroke{Я}	Я	\keystroke{F1}	F1

› **Клавиши, определённые в пакете *keystoke*.**

LCD-дисплей

LCD-дисплеи сейчас встроены даже в кофемолки. Они легко узнаваемы, поэтому нет необходимости копировать их вид в документацию с помощью фотографий – достаточно нарисовать что-то похожее. Изобразить вид дисплея можно с помощью пакета *lcd*.

```
\definecolor{darkgreen}{rgb}{0.22,0.26,0.19}
\definecolor{lightgreen}{rgb}{0.05,0.97,0.55}
\LCDcolors{darkgreen}{lightgreen}
\centering
\LARGE\textLCD{12}|Linux Format|\\[2mm]
\LCDcolors{lightgreen}{darkgreen}
\small\textLCD{12}|Linux Format|
```



Для определения цветов используется макрос \definecolor из пакета *color*. Команда \LCDcolors формирует цвет букв и фона, а макрос \textLCD выводит LCD-подобный текст на экран. \textLCD понимает стандартные команды изменения размера шрифта, поэтому его можно использовать совместно с обычным текстом внутри абзаца.

По умолчанию определены только латинские буквы, цифры и некоторые из стандартных символов. Для определения других символов можно воспользоваться макросом \DefineLCDchar. Макросу передаётся имя символа и битовая маска, определяющая картинку 5x7 точек. Имя символа может быть однобуквенным, тогда соответствующая буква замещается новым рисунком, или многобуквенным, тогда созданный рисунок кодируется указанным словом в фигурных скобках. Другие размеры матрицы в пакете отсутствуют, но при желании его вполне можно доработать.

```
\DefineLCDchar{yo}{01010000001111110001111001000111111}
```

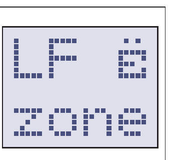
› **Определяем букву «Ё» для LCD.**

» **Месяц назад** Мы научились вставлять графику в документы *LaTeX*.

программный код

Для эмуляции дисплея используется команда \LCD. В качестве обязательных параметров ей передаётся число строк и число столбцов, за которыми следует содержание строк, разделённых каким-то разделителем. В приведённом примере в качестве разделителя используется вертикальная черта, но на её месте мог бы быть любой символ:

```
\DefineLCDchar{yo}{01010000001111110001111001000111111}
\definecolor{lightblue}{rgb}{0.9,0.91,0.99}
\definecolor{darkblue}{rgb}{0.14,0.2,0.66}
\LCDcolors{darkblue}{lightblue}
\LCDframe
\setlength{\LCDunitlength}{1.5mm}
\LCD{2}{4}|LF {yo} |
          |zone |
```



Битовые поля

Для описания сетевых протоколов, а так же для бинарных форматов данных удобнее всего представить последовательность битов графически, то есть в виде таблицы. Это специализация пакета *bytefield*. В пакете определено одноимённое окружение bytefield, в качестве обязательного аргумента которому передаётся ширина таблицы в битах:

```
\begin{bytefield}{«битовая ширина поля»}
  «битовые поля»
\end{bytefield}
```

В окружении bytefield работают команды \wordbox и \bitbox, которые формируют поля, занимающие всю ширину таблицы или только часть её, соответственно:

```
\wordbox[«рамка»]{«число строк»}{«текст»}
\bitbox[«рамка»]{«число занимаемых битов»}{«текст»}
```

Необязательный параметр «рамка» позволяет сформировать обрамление для текущего битового поля. Значение по умолчанию [lrtb] означает, что рамка рисуется со всех сторон поля: l – слева, r – справа, t – сверху и b – снизу. Строки разделяются двойной обратной чертой \\.

Формат пакета сетевого протокола UDP[1] можно описать примерно следующим образом:

```
\begin{bytefield}{32}
\bitheader{0,15,16,31}\\
\wordgroupr{Заголовок}
\bitbox{16}{Порт отправителя}\bitbox{16}{Порт получателя}\\
\bitbox{16}{Размер}\bitbox{16}{Контрольная сумма}
\endwordgroupr\\
\wordbox[lrt]{1}{Данные}\\
\skippedwords\\
\wordbox[lrb]{1}{до 65{.}527 байт}
\end{bytefield}
```

Кроме уже упомянутых команд создания полей при описании формата UDP использовалась команда нумерации столбцов \bitheader, конструкция для создания группы \wordgroupr и макрос \skippedwords для формирования «разрыва».

В качестве обязательного аргумента команде \bitheader передаётся список нумеруемых битов, при этом можно передавать диапазоны чисел, например, **{0-31}**. В пакете определены два окружения для группировки битовых полей, \wordgroupr и \wordgroupl – отличие этих команд в том, что для первой заголовок группы вводится справа, а для второй – слева. Для более подробной информации следует обратиться к документации пакета.

0 … 15	16 … 31	
Порт отправителя	Порт получателя	} Заголовок
Размер	Контрольная сумма	
Данные		
до 65.527 байт		

› **Формат пакета UDP.**

Форматирование кода

LaTeX может использоваться не только для набора математики. Хотя набор математики, безусловно, вершина типографского искусства, существует масса задач со сравнимой сложностью форматирования. Представление исходных текстов программ – это весьма непростое занятие, особенно если хочется сделать код читаемым.

verbatim

Самый простой способ включить код в текст – это заключить его в стандартное окружение verbatim, которое просто выводит данный текст на печать с отступами пользователя. При этом исчезает возможность делать акценты в избранных местах.

Небольшие вставки можно делать с помощью команды \verb!текст!. Сразу после \verb должен идти группирующий символ (в данном случае, «!»), который указывает окончание действия команды. Группирующий символ может быть любым, кроме пробела или звёздочки «*».

Пакет *verbatim* из коллекции *tools* переопределяет стандартную команду так, что внутри окружения можно вставлять тексты неограниченного размера. Кроме этого, пакет предоставляет команду \verbatiminput, которой в качестве основного аргумента можно передать имя внешнего файла.

Кроме упомянутых окружения и макросов определены такие же, но со звёздочкой в конце имени, то есть окружение verbatim* и команды \verb* и \verbatiminput*. *-форма отличается от базовой тем, что все пробелы визуализируются.

```
\begin{verbatim*}
\textbf{verbatim} в         \LaTeX{}
\end{verbatim*}
```

```
\textbf{verbatim}␣в␣␣␣␣␣␣␣␣␣\LaTeX{}
```

Стандартный пакет *altt* – это почти то же самое, что и *verbatim*, но с возможностью использовать внутри окружения команды *LaTeX*. Правда, шрифт в любом случае остаётся фиксированной ширины, как для печатной машинки.

```
\begin{alltt}
 \textbf{altt} в            \LaTeX{}
\end{alltt}
```

altt в LaTeX

Гораздо более разнообразные средства управления выводом неформатированного текста предоставляет пакет *fancyvrb*. За подробной информацией следует обратиться к документации пакета.

»

[2] User Datagram Protocol — это сетевой протокол для передачи данных в сетях IP.

» listings

Пакет *listings* специализируется на оформлении программного кода. К пакету прилагается подробнейшая документация[2]. С помощью команд пакета можно включить как небольшие кусочки кода, так и целые его сегменты, ну и, естественно, файлы.

Для загрузки пакета *listings* необходимо добавить в заголовок следующие инструкции:

Listing 1. Заголовок listings

```
\usepackage{listings}
% подгружаемые языки — подробнее в документации listings
\lstloadlanguages{[LaTeX]TeX,bash,MetaPost,Fortran,Perl,C++,make}
% включаем кириллицу и добавляем кое-какие опции
\lstset{language=[LaTeX]TeX, % выбираем язык по умолчанию
        extendedchars=true,  % включаем не латиницу
        escapechar=|,        % |«выпадаем» в LaTeX|
        frame=tb,            % рамка сверху и снизу
        commentstyle=\itshape, % шрифт для комментариев
        stringstyle=\bfseries} % шрифт для строк
```

Сразу после загрузки пакета рекомендуется «подгрузить» используемые в тексте языки программирования[3] с помощью макроса \lstloadlanguages. В квадратных скобочках перед названием языка можно указать желательный диалект.

Команда \lstset позволяет устанавливать значения по умолчанию. Некоторые из полезных умолчаний перечислены ниже:

» Для того, чтобы можно было печатать кириллицу, например в комментариях, следует определить переменную extendedchars=true[4].

» Опция escapechar позволяет при наборе кода пользоваться услугами *LaTeX* напрямую. Всё, что находится между выбранными символами, обрабатывается средствами *LaTeX*. Естественно, если выбранный символ (в данном случае «|») используется в отображаемом языке, то могут возникнуть проблемы при компиляции. Для того, чтобы обнулить escapechar, достаточно ничего не писать за знаком равно при следующем переопределении.

» Инструкция frame=<POSITION> позволяет рисовать рамку вокруг сегмента кода. На вход принимаются буквы t – обрамление сверху, b – снизу, l и r – слева и справа, соответственно. В случае frame=trbl будет нарисована простейшая одинарная рамка. Опция frame= эквивалентна отказу от обрамления. Если вместо прописных букв указать заглавные frame=TRBL, то рамка будет двойная. В пакете есть возможность сделать рамки посложнее.

Все команды, определённые в пакете *listings*, начинаются с префикса lst. Команда для включения небольших кусочков кода \lstinline!код! аналогична по действию команде \verb!текст!.

Сегмент кода оформляется с помощью окружения lstlisting:

```
\begin{lstlisting}[language=Perl,
 caption={Включение сегмента кода}]
# проверка для перезаписи
if (open(CHECK,"<$file")) {
 $cmd=$term->
  readline("Overwrite (yes/NO): ");
 if (lc($cmd) ne "yes") {die;}
 close(CHECK);}

\end{lstlisting}
```

Listing 1. Включение сегмента кода

```
# проверка для перезаписи
if (open(CHECK,"<$file")) {
 $cmd=$term->
  readline("Overwrite␣(yes/NO):␣");
 if (lc($cmd) ne "yes") {die;}
 close(CHECK);}
```

Необязательный параметр может принимать опции, специфичные для оформления этого куска кода. Например, опция language позволяет установить язык программирования отличный от выбранного по умолчанию, а caption создаёт подпись к фрагменту кода.

Файлы можно включать с помощью команды \lstinputlisting:

```
%установка значений по умолчанию
\lstset{numbers=left, language=MetaPost,
     backgroundcolor=\color{yellow},
     frame=shadowbox, rulesepcolor=\color{black}}
%вставка файла
\lstinputlisting[firstline=16, lastline=24,
  emph={forsuffixes,text,bpath},emphstyle={\color{red}},
  emph={[2]fill,unfill},emphstyle={[2]\bfseries\underbar},
]{intro.mp}
```

```
vardef drawshadowed(expr dx,dy)(text t) =
  fixsize(t);
  forsuffixes s=t:
    fill bpath.s shifted (dx,dy);
    unfill bpath.s;
    drawboxed(s);
%     draw pic(s) withcolor red; %цвет текста
  endfor;
enddef;
```

С помощью опций firstline и secondline можно указать диапазон строк, который следует вывести. В зависимости от выбора языка форматирование существенно меняется. Инструкция numbers=left нумерует строки слева.

Для работы с цветами лучше загрузить уже упоминавшийся ранее пакет *color*. Цвета хороши для выделения каких-то ключевых слов и подложки, за которую отвечает опция backgroundcolor. Возможности для определения своих «словариков» предоставляет опции emph=<список ключевых слов>. В начале списка может идти его метка в квадратных скобках, таким образом, можно поддерживать несколько списков одновременно. С помощью опции emphstyle можно определить способ выделения ключевых слов.

Обычно код располагается прямо по месту основного текста, так как обсуждение «исходников» можно производить в самом коде, благо есть комментарии. При желании можно воспользоваться опцией float, чтобы из фрагмента кода получился полноценный «плавающий» объект.

Пакет поддерживает свыше сотни распространённых языков программирования и разметки (с учётом диалектов), так что вам, скорее всего, не придётся определять свой язык с помощью инструкции \lstdefinelanguage. Но если очень хочется, то и это возможно.

Представление алгоритмов

Собственно говоря, это именно то, ради чего Д.Э. Кнут и создал TeX. Поэтому пакеты для облегчения записи алгоритмов в *LaTeX* существовали с самого его рождения. На текущий момент, даже число стандартных пакетов, подпадающих под эту тематику, больше десятка. Здесь рассмотрена только малая часть из них.

algorithm

Пакет *algorithm* ориентирован на описание алгоритмов, а не на представление кода. Это позволяет отрешиться от форматирования и сосредоточиться на основной задаче. Пакет определяет окружение algorithmic. Для использования в преамбуле следует загрузить одноимённый стиль.

```
\begin{algorithmic}[1]
\IF{\(i\leqslant0\)} \STATE \(i\gets1\) \ELSE
 \IF{\(i\geqslant0\)} \STATE \(i\gets0\)
 \COMMENT{смысла в этом алгоритме не ищите}
 \ENDIF
\ENDIF
\ENSURE \(i\geqslant0\)
\FORALL{\(\xi \in \mathcal{A}\)}
 \STATE \(\mathcal{B}\gets\xi^2\)
\ENDFOR
\RETURN \(\mathcal{B}\)
\end{algorithmic}
```

if $i \leqslant 0$ **then**
  $i \leftarrow 1$
else
  if $i \geqslant 0$ **then**
    $i \leftarrow 0$ {смысла в этом алгоритме не ищите}
  end if
end if
Ensure: $i \geqslant 0$
for all $\xi \in \mathcal{A}$ **do**
  $\mathcal{B} \leftarrow \xi^2$
end for
return $\mathcal{B}$

› Пример использования пакета *algorithm*.

[2] Следует поискать файл listings.pdf.

[3] Текущая версия пакета 1.3c поддерживает следующие языки (в скобках указаны диалекты): ABAP, ACSL, Ada (83, 95), Algol (60, 68), Ant, Assembler (x86masm), Awk (gnu, POSIX), bash, Basic (Visual), C (ANSI, Handel, Objective, Sharp), C++ (ANSI, GNU, ISO, Visual), Caml (light, Objective), Clean, Cobol (1974, 1985, ibm) Comal 80, csh, Delphi, Eiffel, Elan, erlang, Euphoria, Fortran (77, 90, 95), GCL, Gnuplot, Haskell, HTML, IDL (empty, CORBA), inform, Java (empty, AspectJ), JVMIS, ksh, Lisp (empty, Auto), Logo, make (empty, gnu), Mathematica (1.0, 3.0), Matlab, Mercury, MetaPost, Miranda, Mizar, ML, Modula-2, MuPAD, NASTRAN, Oberon-2, OCL (decorative, OMG), Octave, Oz, Pascal (Borland6, Standard, XSC), Perl, PHP, PL/I, Plasm, POV, Prolog, Promela, Python, R, Reduce, Rexx, RSL, Ruby, S (empty, PLUS), SAS, Scilab, sh, SHELXL, Simula (67, CII, DEC, IBM), SQL, tcl (empty, tk), TeX (AlLaTeX, common, LaTeX, plain, primitive), VBScript, Verilog, VHDL (empty, AMS), VRML (97), XML, XSLT.

[4] Если это не сработает, то необходимо обновить пакет до последней версии или сменить дистрибутив LaTeX на более подходящий.

Если необязательный аргумент определён, то осуществляется нумерация строк. Если аргумент равен **1**, то нумеруются все строки, если **2** – то каждая вторая, а далее по индукции.

Команда \STATE определяет простое утверждение. Условный оператор представлен командами \IF{<условие>}, \ELSIF{<условие>}, \ELSE и \ENDIF. Циклы представлены операторами \FOR и \FORALL, которые закрываются командой \ENDFOR. Аналогично присутствуют пары \WHILE{<условие>} – \ENDWHILE, \REPEAT – \UNTILL{<условие>} и бесконечный цикл \LOOP – \ENDLOOP. Кроме уже перечисленных конструкций определены предварительное условие для корректного выполнение алгоритма \REQUIRE, постусловие, которое должно выполняться при корректной работе алгоритма, \ENSURE, возвращение результата \RETURN, промежуточная печать \PRINT и комментарий \COMMENT.

Собственно говоря, всё. Псевдокод автоматически разбивается на строки и форматируется в соответствии с общепринятыми представлениями. Очевидно также, что навыки набора математики будут здесь очень кстати. Подробности по настройке пакета следует выяснять в документации к нему: **algorithms.pdf**.

Для того, чтобы сделать из объекта algorithmic «плавающий объект» можно воспользоваться окружением algorithm, для его использования следует загрузить одноимённый стиль в преамбуле. Внутри algorithm можно использовать команды \caption и \label.

Клоны algorithms

С использованием имеющихся наработок пакета *algorithms* был создан *algorithmicx*. Этот пакет предоставляет более расширенный набор команд. Кроме того, пользователю предоставляются команды, с помощью которых можно формировать свои алгоритмические конструкции. Автор также предоставил вариант форматирования отступов, принятый в Pascal, что позволяет относительно легко приводить программы на этом языке к виду, годному для красивой распечатки. Пакет работает с теми же окружениями, что используюся в пакете *algorithms*. Это приводит к их несовместимости.

Решение схожей функциональности предоставляет пакет *algorithm2e*. Форматирование С-подобно. Предоставлен избыточный набор конструкций и возможность самому создавать новые структуры. Есть зачатки локализации. Пакет использует окружение algorithm, Это приводит к несовместимости как с пакетом *algorithms*, так и с пакетом *algorithimcx*.

clrscode

Пакет *clrscode* представляет возможность набирать псевдокод, как это делали авторы книги «Алгоритмы: построение и анализ» Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест и Клиффорд Штайн[5]. Для работы с пакетом необходимо загрузить одноимённый стиль. Это прекрасный пример того, как можно адаптировать *LaTeX* для создания книг по программированию.

```
\begin{codebox}
\Procname{
     $\proc{Сортировка методом вставок}$}
\li \For $j \gets 2$ \To $\id{length}[A]$
\li \Do $\id{key} \gets A[j]$
\li $i \gets j-1$
\li \While $i > 0$ and $A[i] > \id{key}$
\li \Do $A[i+1] \gets A[i]$
\li $i \gets i-1$ \End
\li $A[i+1] \gets \id{key}$ \End
\end{codebox}
```

Сортировка методом вставок

for $j \leftarrow 2$ **to** $length[A]$
    do $key \leftarrow A[j]$
        $i \leftarrow j-1$
        while $i > 0$ and $A[i] > key$
            do $A[i+1] \leftarrow A[i]$
                $i \leftarrow i-1$
        $A[i+1] \leftarrow key$

› **Пример использования пакета *clrscode.***

pseudocode

Профессора Дональд Л. Крехер [Donald L. Kreher] и Дуглас Р. Стинсон [Douglas R. Stinson] написали книгу «*Combinatorial Algorithms: Generation, Enumeration and Search*». Специально для представления псевдокода в этой книге они создали пакет, который так и назвали: *pseudocode*. Дональд Л. Крехер использовал одноимённое окружение и в своей следующей книге по алгоритмам, выпущенной уже в 2005 году. Пакет поддерживается до сих пор.

```
\begin{pseudocode}{C2F\_таблица}
                  {\text{от}, \text{до}}
\PROCEDURE{C2F}{c}
\COMMENT{Преобразование
   $^\circ$C$\to$$^\circ$F}\\
f \GETS {9c/5} + 32\\
\RETURN{f}
\ENDPROCEDURE
\MAIN
x \GETS \text{от} \\
\WHILE x \leqslant \text{до} \DO
\BEGIN
\OUTPUT{x, \CALL{C2F}{x}}\\
x \GETS x+1
\END
\ENDMAIN
\end{pseudocode}
```

Algorithm 1.1: C2F_таблица(от, до)

procedure C2F(c)
  comment: Преобразование °C→°F
  $f \leftarrow 9c/5 + 32$
  return (f)

main
  $x \leftarrow$ от
  while $x \leqslant$ до
    do $\begin{cases} \textbf{output } (x, \text{C2F}(x)) \\ x \leftarrow x+1 \end{cases}$

› **Пример использования пакета *pseudocode.***

К сожалению, в книгах по *LaTeX* редко рассматриваются структуры, полезные для представления программных текстов или псевдокода. Здесь я попытался восполнить этот зияющий пробел. Тема настолько обширна, что разрабатывать её можно почти бесконечно. *LaTeX* сам по себе код, поэтому программистам, по идее, должно быть уютно в его окружении. LXF

LaTeX и контроль версий

Исходный текст *LaTeX* тоже представляет собой код. И как всякий код, он достоин жить в системе контроля версий. Часто бывает любопытно узнать текущую версию документа и время его последнего обновления. Если в качестве системы контроля версий используется *Subversion*, то для начала следует загрузить пакет *svn*[6].

```
\usepackage{svn}
\SVN $Date$
\SVN $Rev$
```

При этом в текст следует добавить метки, предваряемые командой \SVN. Для интерполяции меток в системе *Subversion* при обновлении файла следует выполнить команды вида:

```
> svn propset svn:keywords "Date Rev" «имя файла»
> svn commit -m "интерполяция меток"
```

При этом *svn* передаётся информация о том, какие именно метки требуется обновлять при выполнении commit. В данном случае, это метки Date и Rev — дата и номер ревизии, соответственно. Более подробную информацию можно получить с помощью команды

```
> svn help propset
```

Команда \SVN $Date$ определяет команды \SVNDate и \SVNTime, ответственные за календарную дату и время. Все остальные команды вида \SVN $Keyword$, где Keyword — одна из интерполируемых меток *svn*, определяют команды \SVNKeyword.

После интерполяции метки будут выглядеть примерно следующим образом (*см. рисунок*).

Схожую функциональность предоставляет пакет *svninfo*.

```
\SVN $Date: 2006-11-25 21:02:20 +0600 $
\SVN $Rev: 265 $
Документ обновлён \SVNDate\ \SVNTime

Текущая версия \SVNRev
```

Документ обновлён 25 ноября 2006 г. 21:02:20

Текущая версия 265

[5] Introduction to algorithms, Second Edition. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein.

[5] Если же в вашем проекте используется CVS (Concurrent Versions System), то следует воспользоваться пакетом *rcs*. За подробностями следует обратиться к документации пакета.

Через месяц Вы узнаете всё об оформлении абзаца и страницы.

Новая серия! Свободный инструментарий для трехмерного моделирования

Курс молодого

ЧАСТЬ 1: Сегодня **Александр Супрунов** проведет мастер-класс по пакету *Blender* – и снабдит вас «джентльменским минимумом», который неоднократно пригодится вам в дальнейшем.

От редакции

Предлагаемая вашему вниманию серия учебников *Blender* – новое слово в долгой истории *Linux Format*. Дело не только в том, что уникальный по-своему пакет *Blender* не избалован избытком документации (о чем неоднократно говорили наши британские коллеги), но и в том, что эту серию будут писать сразу два человека – Александр Супрунов и Андрей Прахов. Две головы лучше, чем одна, и мы надеемся, что опыт двух экспертов поможет вам лучше понять всю многогранность *Blender*. Приятного чтения!

Начиная с этого занятия мы будем учиться создавать трехмерные модели и поможет нам в этом программа *Blender*.

Blender, некогда коммерческий пакет для моделирования и анимации трехмерных объектов, разработанный компанией NaN (Not a Number) и использовавшийся художниками знаменитой голландской анимационной студии NeoGeo для создания передовых проектов, был впервые представлен широкой публике на конференции Siggraph в 1999 году, где вызвал огромный интерес. Сегодня *Blender* является свободным ПО и распространяется на условиях GNU GPL. Примечательно, что не последнюю роль в этом сыграло сообщество: права на исходные тексты *Blender* были приобретены у первоначального владельца за 100 000 евро, которые были собраны в виде пожертвований в рамках акции «*Free Blender*», причем произошло это в довольно короткий срок: всего за семь недель. *Blender* начал новую жизнь как открытый проект 13 октября 2002 года и с тех пор его можно найти по адресу: www.blender.org.

Не станем более вдаваться в историю, но отметим – к вам в руки попал уникальный и невероятно мощный инструмент для создания трехмерных моделей, способный выполнить практически любую поставленную перед ним задачу.

Одной из главных изюминок *Blender* является интерфейс. Он совершенно уникален и ориентирован на продуктивную работу. С другой стороны, это означает, что ваши привычки и навыки работы с другими программами здесь не слишком помогут. *Blender* чем-то похож на *Emacs* – поначалу кажется странным и непонятным, а спустя некоторое время – очень удобным и эффективным.

Мы не будем специально касаться установки *Blender* – он включен во многие современные дистрибутивы. Убедитесь лишь, что вы используете версию не ниже 2.42a – в противном случае, кое-что из описанного в этой и последующей статьях цикла может для вас не сработать. Если версия *Blender* в вашем дистрибутиве подустарела – загрузите новую с сайта проекта или возьмите ее с нашего DVD. Официальная бинарная сборка *Blender* 2.42a должна заработать на любой системе с *glibc 2.3.2* и выше.

Первое впечатление

Запустите *Blender*. Вы увидите окно, разделенное по горизонтали на две области (**Рис. 1**).

В дальнейшем мы сможем добавить дополнительные области, сообразуясь с нашими потребностями.

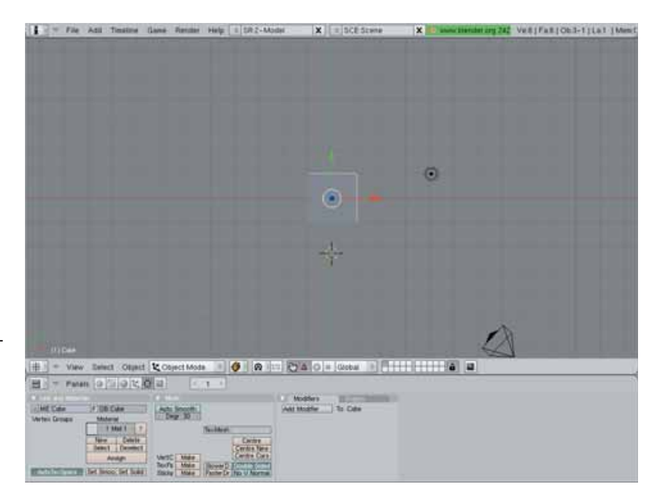


Рис. 1. *Blender* во всей красе. Обратите внимание на отсутствие рамок – по умолчанию, программа занимает весь экран.

В верхней части окна расположен куб, вид спереди. Вы также можете заметить камеру (в форме пирамиды), направленную на куб и источник света (круг с пунктирной линией, показывающей направление).

Давайте освоимся на новом месте. Переместите курсор мыши в одно окно с кубом и, зажав среднюю кнопку (она же колесико), подвигайте мышью – это докажет сомневающимся, что перед нами никакой не квадрат, а самый настоящий куб. Переключение видов осуществляется клавишами на дополнительной цифровой клавиатуре:

- 7 – вид сверху.
- 1 – вид спереди.
- 3 – вид сбоку.
- 0 – вид из камеры.

Чтобы вернуться к виду спереди, нажмите клавишу 1.

При создании объекта часто возникает необходимость приблизить, отдалить или повращать модель. Для этого можно использовать колесо прокрутки мыши: вращение вперед – приближение, назад – отдаление. Чтобы «протащить» экран влево или вправо (и, таким образом, увидеть объекты, ранее не попадавшие в зону видимости), переместите указатель мыши, зажав клавишу Shift или среднюю кнопку мыши.

Объекты на экране можно выделять – это необходимо, если над ними требуется выполнить какое-то действие. Для выделения объекта необходимо навести на него курсор и щелкнуть правой кнопкой мыши.

Теперь давайте потренируемся. У вас на экране есть три объекта – куб, камера и источник освещения («лампа»). Отдалите вид, вращая колесико мыши «на себя», так, чтобы они были видны одновременно. Затем выделите «лампу», камеру, куб. Получилось? Движемся дальше!

Действия над объектами

Думаю, вы уже довольно уверенно управляетесь со сценой в целом, так что теперь пришло время спуститься на уровень ниже – к отдельным объектам. Что можно с ними делать? Первое, что приходит на ум – дви-

бойца





Комментарий Андрея Прахова

Навигация и клавиатура

Blender обладает широчайшими возможностями в области навигации по экрану. Помимо указанных горячих клавиш (0, 1, 3, 7 на дополнительной клавиатуре), можно активно использовать и остальные цифры, причем выполняемые ими функции зависят как от состояния клавиши NumLock, так и от клавиш-регистров. При включенном NumLock вы можете вращать отображаемую область в любых направлениях. Клавиши 2, 8 служат для вертикальной ротации, 4, 6 – для горизонтальной. С помощью + и – вы, как и при использовании колесика мыши, задействуете механизм увеличения (уменьшения). «Точка» позволит отцентрировать и разместить выделенный объект так, чтобы при максимально большом увеличении он смог разместиться на экране. Очень полезна клавиша Enter. Если вы ненароком умудрились раскрутить колесико мыши так, что объекты испарились в неведомой дали – нажмите ее и все вернется на место. Для переключения из ортографической проекции в перспективу служит клавиша 5.

Клавиатура ведет себя совсем по-другому, если удерживать нажатым Ctrl. В этом случае клавиши 2, 4, 6, 8 служат для перемещения всей рабочей области в соответствующих направлениях. И, наконец, если вы разработчик игр и используете *Blender* для создания ландшафта к какому-нибудь DOOM-X, то вам будет полезен режим «свободной камеры» – нажмите комбинацию Shift+F. Играйте!

гать! Давайте посмотрим, что для этого нужно.

Прежде всего, намеченный для перемещения объект нужно выделить – при этом вокруг него появится розовый ободок. Для перемещения объекта используется горячая клавиша – G. Нажмите ее (при необходимости, переключите клавиатуру на английскую раскладку) и двигайте мышью (или используйте клавиши управления курсором), наблюдая, как перемещается объект. Зафиксировать изменения можно левой кнопкой мыши, отменить – клавишей Esc.

Иногда бывает необходимо передвинуть объект только по одной из трех осей координат (**X**, **Y**, **Z**). Для этого следует воспользоваться цифровой панелью «Transform Properties» (**рис. 2**), вызываемой по клавише N. Повторное нажатие на N удалит панель с экрана.

Перемещая объект, вы можете следить за изменением его координат в панели «Transform Properties». Обратите внимание, что справа от каждого значения изображен открытый замочек. «Защелкнув» его левой кнопкой мыши, можно запретить изменять координаты объекта по данной оси. Например, если вам необходимо перемещать объект вдоль оси X, защелкните замочки на LocY и LocZ. Затем нажмите горячую клавишу – G – и убедитесь, что объект перестал двигаться в двух других направлениях. Чтобы вернуть ему утраченные степени свободы, отомкните замочки повторным щелчком мыши. Аналогичным образом можно ограничить трансформации увеличения и вращения по определенным осям.

Рис. 2. Панель «Transform properties».

Проба пера

Пожалуй, настало время произвести первый рендеринг – то есть изобразить находящиеся на сцене объекты в соответствии с заданными им свойствами, освещением и т.д.

Выполнить его довольно просто. Нажмите клавишу F12, и через несколько секунд вашему взору предстанет готовая сцена с трехмерным кубом.

Сохраните ее в виде растрового изображения, нажав F3, вписав в появившемся окне название будущего графического файла с расширением **.jpg** (например, **cube.jpg**) и щелкнув по кнопке Save Jpeg.

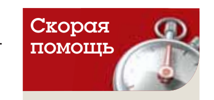
Но что это? Картинка, получившаяся при рендеринге, отличается от той, что мы видели на экране. Чтобы увидеть такую же проекцию, что и в конечном результате, необходимо включить ракурс Вид из камеры, что, как мы уже знаем, выполняется нажатием на клавишу 0 (ноль) на дополнительной цифровой клавиатуре. Вернуться к виду спереди можно, нажав клавишу 1.

Постоянно переключаться в разные вид ы во время подготовки сцены для рендеринга не слишком удобно. Здесь нам опять приходит на помощь приятная продуманность интерфейса *Blender*. Ничто не мешает нам создать еще одно окно, которое будет отображать вид из камеры!

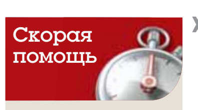
Подведите курсор к верхней кромке рабочей части экрана и нажмите правую кнопку мыши. Появится меню:

- » Split Area – разделить область;
- » Join Areas – объединить области;
- » No header – удалить заголовок окна.

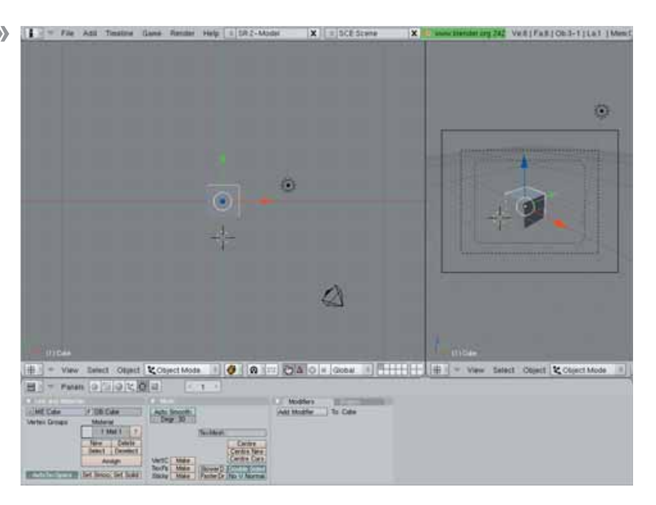
Выберите Split Area. Появится вертикальная линия. Передвиньте ее, сделав новое окна желаемого размера и щелкните левой кнопкой мыши. В результате этих действий экран будет разделен на две части (**рис. 3**). Переместите курсор мыши на новый экран и нажмите клавишу 0 – теперь он отображает вид из камеры.



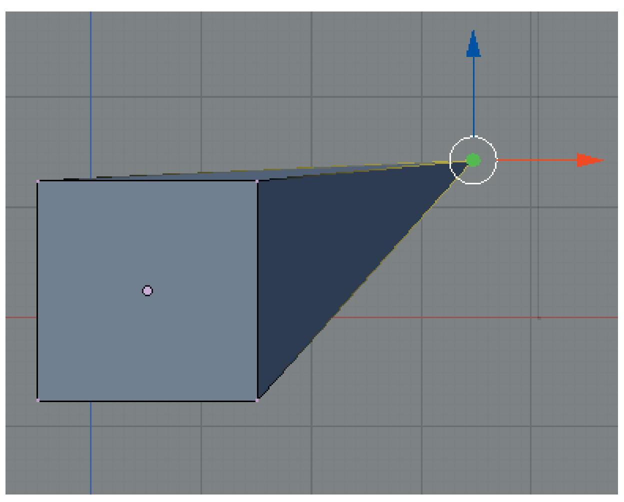
Для выделения двух и более объектов нужно удерживать клавишу **SHIFT**.



Выбор пунктов меню доступен не только курсором мыши, а и нажатием цифровых клавиш, соответствующих пункту. Самый верхний пункт имет номер **1**, второй – **2** и тд.



(Рис. 3) Легким движением мыши, рабочая область превращается… в две рабочих области.



(Рис. 4) Куб меняет форму.

Чтобы вернуть все, как было, щелкните по вертикальной полосе между окнами правой кнопкой мыши и выберите Join Areas (объединить области) – после чего появится стрелка, которой следует указать область для удаления.

Все, что мы пока делали, происходило в режиме Object Mode – о чем можно было догадаться, взглянув на панель внизу экрана. Этот режим позволяет управлять объектами – перемещать, вращать их и т.д. Но что делать, если нам требуется внести изменения в сам объект? Для этого необходим другой режим – Edit Mode.

Режим правки

Режим правки включается нажатием на клавишу Tab, либо выбором соответствующего пункта из выпадающего списка Mode в нижней части экрана.

Как видите, куб изменился. На вершинах появились точки, а сам объект «опоясался» желтой рамочкой, означающей, что он выделен. Для снятия выделения можно использовать клавишу A.

Повращайте куб, зажав среднюю кнопку мыши – вы увидите, что на каждом из углов куба появилась розовая точка. Воздействуя на нее, мы можем менять форму куба.

Переключитесь на вид сбоку (клавиша 3) и щелкните правой кнопкой мыши по одной из точек. Она изменит цвет на желтый, сигнализируя о выделении. Как вы, надеюсь, помните, горячая клавиша G позволяет перемещать объект. Нажмите ее и отодвиньте точку в сторону, наблюдая за деформацией куба. Зафиксируйте результат левым щелчком мыши. Подправьте модель, переключаясь на разные виды. Отменить сделанные безобразия можно, нажав Ctrl-Z или U, а вернуть изменения – нажав Shift-U.

Чтобы выделить не только точку, которой коснулся курсор, но и все точки, лежащие на данной оси, потребуется другой инструмент – блочное выделение. Вызывается он горячей клавишей B и имеет два режима. Мы продемонстрируем его использование, выделив верхнюю грань куба:

- Снимите выделение с куба клавишей A.
- Нажмите B и активируйте инструмент блочного выделения.
- Курсор примет вид перекрестия. Наведите его чуть выше и левее куба и, зажав левую клавишу мыши, «протащите» прямоугольник, выделив верхние точки куба.
- Примените инструмент перемещения, нажав клавишу G и вытянув куб вверх. Чтобы перемещение происходило строго вертикально, удерживайте клавишу Ctrl.

Помимо блочного выделения, существуют еще два инструмента, которые используются очень и очень часто. Речь идет о вращении и масштабировании.

Начнем с вращения. Для него предусмотрена горячая клавиша R (Rotate).

Вращать можно как отдельные вершины, так и весь объект. Все зависит от текущего режима – Object Mode или Edit Mode. В режиме Object Mode (для перехода в него из режима правки следует опять нажать клавишу Tab) мы не видим отдельных вершин и не можем манипулировать ими. Убедитесь, что объект выделен (розовая каемка на месте) и нажмите клавишу R, а затем – подвигайте мышью. Вы увидите, как вращается объект. Здесь есть одна хитрость: вы можете конкретизировать ось, вокруг которой происходит вращение. Для этого после нажатия R нажмите одну из следующих клавиш:

- **Z** – для вращения по оси **Z**,
- **X** – для вращения по оси **X**,
- **Y** – для вращения по оси **Y**

и смело беритесь за мышь.

Перейдем к следующему инструменту – Scale или «Изменение размера». Вызывается он горячей клавишей S. Ось для масштабирования задается аналогично Rotate. Если ось не задана, размеры объекта изме- »

Комментарий Андрея Прахова



Маленькие хитрости

Строго говоря, клавиша Ctrl способна на гораздо большее, чем простое выравнивание при перемещении вдоль какой-либо из оси. Если вам необходим точный контроль над изменением параметров перемещения, поворота или масштабирования, воспользуйтесь ею! Например, активируйте режим масштабирования (S) и, удерживая нажатой Ctrl, потяните мышь в любом направлении. Почувствовали разницу? При перемещении, масштабировании и выдавливании параметры меняются на **0.1000**, при повороте – на 5 градусов. Эффекта выравнивания по одной из осей также можно добиться, если удерживать нажатой среднюю кнопку мыши. Ось указывается соответствующим направлением движения «грызуна».

И последнее: подобно *Opera*, *Blender* понимает «жесты» мышью. Так, прочертив прямую, вы активируете режим перемещения. Нарисовав круг – режим поворота. Изображение буквы "**V**" позволит вам воспользоваться инструментом масштабирования.

Практикум

В принципе, полученных вами знаний уже достаточно для моделирования несложных объектов. Для подтверждения этого тезиса приведем простейший вариант моделирования гриба. Не будем вдаваться в спецификации или классификации – белый это гриб или поганка, просто очистим сцену.
Для этого надо нажать Ctrl-X и в появившемся окне выбрать пункт Erase All.
Вы увидите пустое окно с кубом посередине. Удалите куб клавишей Delete.



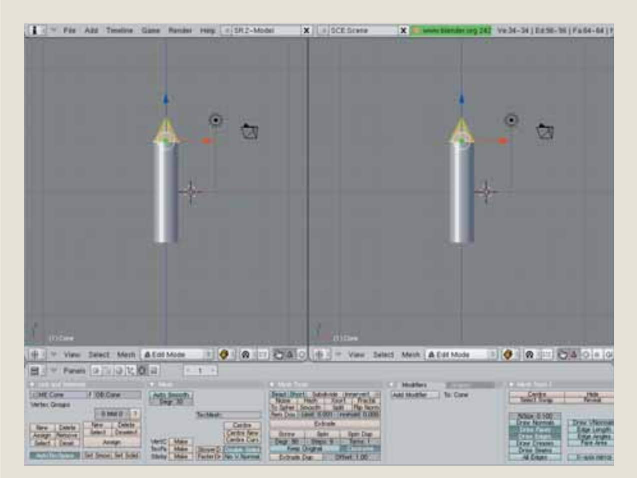
1 Создайте примитив для ножки гриба – Tube (цилиндр)

В Object Mode, из вида «сверху», добавьте сеточный объект Tube: ADD->Mesh->Tube. Blender автоматически переключится в режим «Edit Mode».
Растяните его по оси **Z** до толщины ножки обычного гриба: нажмите клавиши S и Z, а затем двигайте мышью.
Чтобы видеть сделанные изменения в реальном времени, добавьте еще одно окно с видом спереди.



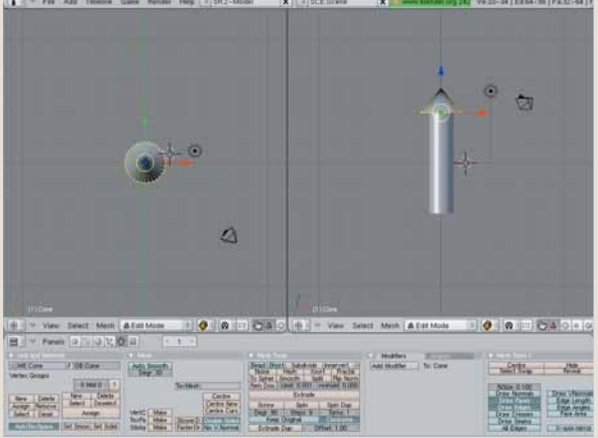
2 Добавьте примитив для шляпки гриба – Cone (конус)

Переключитесь в режим Object Mode клавишей Tab.
Переместите 3D-курсор правее созданного цилиндра.
Добавьте объект Cone: клавиша Space – Add ->Mesh->Cone.



3 Расположите шляпку гриба над ножкой

Переключитесь на вид спереди - 1.
Нажмите G и разместите шляпку над ножкой. Зафиксируйте перемещение щелчком левой кнопки мыши.



4 Растяните шляпку гриба

Снимите выделение со шляпки – клавиша A.
Инструментом выделения (клавиша B) выделите нижний ряд вершин шляпки.
Переключитесь на вид сверху – клавиша 7.
Измените размер, сделав шляпку больше: нажмите клавишу S и двигайте мышью от центра экрана.

Если модель получилась слишком большой, перейдите в режим Object Mode, выделите, удерживая Shift оба объекта (ножку и шляпку) и примените инструмент Scale – S.
Для сохранения результата нажмите Ctrl-S и вместо даваемого по умолчанию имени **untitled.blend** впишите свое – **grib.blend**.

няются пропорционально вдоль всех осей.

Кто-то, возможно, спросит: «Откуда взялся куб при открытии программы *Blender*?».

Этот примитив расположен в центре экрана по умолчанию. Для его удаления следует нажать клавишу Delete и подтвердить в появившемся меню пункт Erase selected Object(s) щелчком мыши. Теперь сцена пуста.

Для упрощения моделирования, *Blender* предоставляет множество как уже готовых примитивов, так и кривых Безье, мета-объектов и т.д. Самыми используемыми являются сеточные объекты (mesh objects).

Доступ к примитивам и большинству других возможностей можно получить из контекстного меню, вызываемому по горячей клавише Space, альтернативное сочетание – Shift-A. Содержимое меню зависит от режима, в котором вы находитесь.

Давайте разберемся, как добавить в сцену новые примитивы. Находясь в режиме Object Mode, нажмите Space и в появившемся меню последовательно выберите пункты Add->Mesh->Plane. Таким образом, в сцену будет добавлена плоскость (plane). Она появится в центре экрана на месте 3D-курсора – перекрестия с красно-белой окружностью. 3D-курсор можно переместить в любое другое место – просто переведите курсор мыши в сторону и щелкните левой кнопкой. При создании объектов, принимайте во внимание из текущий вид (сверху, спереди и т.п.). Проще сразу добавить объект из нужного нам ракурса, чем потом поворачивать его.

Под занавес я припас для вас самое интересное. На сцену выходит тот, кого вы весь урок, думаю, ждали. Великий и прекрасный, совсем не зубастый, пингвин Tux (**рис. 5**)! Эту модель любезно предоставил нам Андрей Прахов, и в следующих статьях цикла он непременно расскажет, как сделать такую же самостоятельно,

Для добавления модели в вашу сцену нажмите Shift+F1 и выберите требуемый файл, имеющий расширение .**blend**, в появившемся диалоге. Вы увидите, что попали в папку, содержащую дочерние категории, в которых хранятся настройки освещения, текстур и т.д. Войдите в директорию **Object** и выберите составляющие модели пингвина. В результате этих действий модель добавится в вашу сцену.

Занавес. Аплодисменты Tux'y! LXF

› (Рис. 5) Финальная сцена первой серии.



» Через месяц Андрей Прахов научит вас рисовать пингвинов.



А давайте

Если вы когда-либо пытались продемонстрировать видеозапись своего отпуска сразу всем своим друзьям и знакомым, то наверняка представляете, как трудно бывает собрать их всех в одном месте. VideoLAN поможет решить эту проблему – просто настройте сервер потокового вещания, и пусть каждый из них насладится видом солнечного пляжа прямо на рабочем месте! **Сергей Яремчук** расскажет, что вам потребуется.



Задачу одновременного просмотра видеоролика большим количеством пользователей можно решить различными способами. Самым простым из них будет открыть доступ к видеофайлу по сети и смотреть его в обычном видеопроигрывателе. Данный метод обладает тремя несомненными достоинствами, а именно: не требует никакого специального программного обеспечения, не зависит от операционной системы, установленной на стороне клиента, сам файл можно просмотреть в любое время. Недостатков же не так много (по количеству), зато их «качественный» вес перетягивает все остальное. Например «живое» видео, полученное с web-камер или со спутникового ТВ таким образом уже не посмотришь. Сервер и канал при большом количестве подключений будет перегружен. Выход – трансляция видеопотоков. Ею мы сейчас сейчас и займемся.

Мы будем использовать VideoLAN (http://www.videolan.org/) – проект, начатый французскими студентами, а сейчас поддерживаемый разработчиками более чем из двадцати стран мира. VideoLAN прост в настройке и обладает всеми необходимыми нам возможностями. С его помощью можно легко организовать трансляцию потоков мультимедиа из различных источников: видеофайлов формата MPEG-1, MPEG-2, MPEG-4 и DivX, цифровых видео и DVD-дисков, цифровых спутниковых и телевизионных каналов и «живых» видеотрансляций, полученных с web-камер. Все это можно осуществить как для отдельного компьютера, так и в небольшой сети или в Интернете. Для трансляции потока MPEG-4 ширина канала должна быть от 0.5 до 4 Мбит/с, для потока MPEG-2, идущего со спутникового или телевизионного канала – от 3 до 4 Мбит/с, а для DVD потребуется от 6 до 9 Мбит/с. Этот проект выделяет еще одна особенность – многоплатформенность. VideoLAN может работать на компьютерах под управлением различных версий GNU/Linux, всех распространенных BSD-систем, Windows, Mac OS X, BeOS, Solaris, QNX. VideoLAN распространяется по лицензии GPL.

Для организации вещания VideoLAN предлагает два приложения:

- *VLS* – сервер VideoLAN, транслирующий (как уже говорилось) потоки из файлов MPEG-1, MPEG-2 и MPEG-4, DVD и VCD, цифровых спутниковые и телевизионных каналов, а также «живое» видео.
- *VLC* – изначально, клиент VideoLAN, способный получать, декодировать и выводить потоки MPEG. Сейчас эта программа может использоваться и как сервер, транслирующий потоки из файлов в формате MPEG-1, MPEG-2 и MPEG-4/DivX, видеодисков и «живое» видео. Помимо этого, VLC является универсальным видеопроигрывателем локальных файлов, поддерживающим большинство форматов.

Для *VLS* отдельно поставляется SAP-сервер. Его назначение – объявление анонсов транслируемых по сети программ (VLC анонсирует себя сам). Клиенты VLC получают эти сообщения и автоматически добавляют объявленные программы в свой плей-лист.

VLC имеет понятный графический интерфейс, позволяющий настроить большинство параметров. Для организации одного потока достаточно компьютера класса Pentium 100 с 32 Мб ОЗУ, жесткий диск, естественно, нужен побольше.

Установка VLC

Перекомпилированные пакеты VLC доступны для большинства дистрибутивов. В системах, базирующихся на APT, все, что касается VLC, можно найти одной командой:

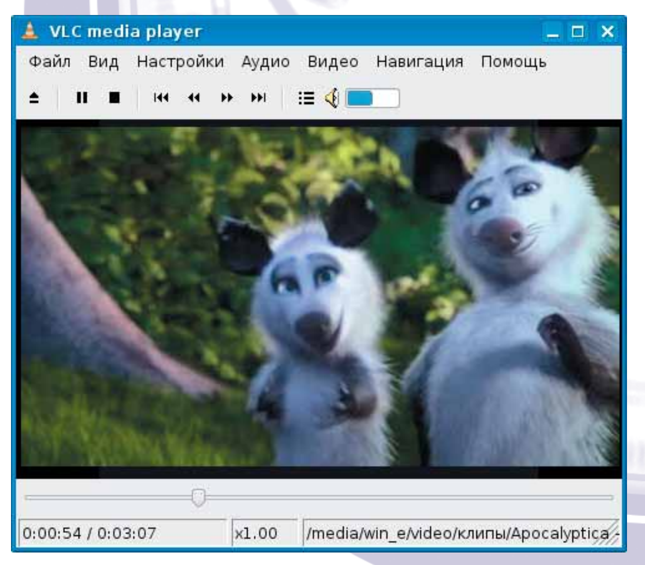
```
$sudo apt-cache search vlc
```

Список будет большим, но, к счастью, все устанавливать не нужно. Например, чтобы установить VLC в минимальной конфигурации в KUbuntu, достаточно ввести:

```
$ sudo apt-get update
$ sudo apt-get install vlc vlc-plugin-alsa #или vlc-plugin-esd для Ubuntu
```

При этом будут установлены все необходимые пакеты, включая зависимости. По умолчанию используется графический интерфейс wxvlc (библиотека wxWidgets/wxGTK), но при желании можно добавить к списку kvlc, qvlc или gnome-vlc и установить интерфейсы для KDE, Qt или GNOME [имейте в виду, что некоторые из них уже официально не поддерживаются,-прим.ред.]. Кроме того, VLC доступен в виде под-

Покажем им всем!



› **Рис. 1**

ключаемого модуля *Mozilla mozilla-plugin-vlc*. Для ALTLinux команда установки выглядит так.

```
$ apt-get install vlc-normal
```

На странице http://www.videolan.org/vlc/ приведены ссылки с информацией по установке VLC в основных дистрибутивах GNU/Linux.

Если вы будете устанавливать VLC вручную, вам понадобится еще несколько дополнительных библиотек:

- *libdvdcss* – если вы планируете читать зашифрованные DVD-диски (кстати, модуль не обращает внимание на зоны);
- *libdvdplay* – если требуется поддержка DVD-меню;
- *libdvbpsi* – если вы планируете читать видеопотоки TS/DVB со спутников или цифрового телевидения;
- *a52dec* – при необходимости декодировать звук в формате AC3 (A52), принятом в мире DVD;
- *ffmpeg*, *libmad* и *faad2* – для чтения файлов MPEG 4/DivX;
- *libogg* и *libvorbis* – для поддержки звукового формата Ogg Vorbis;
- *lirc* – для управления с помощи пульта ДУ

Ключ на старт!

Запустить VLC можно как из основного меню рабочего стола, в которое встраивается ярлык программы, так и с консоли, командой vlc. В результате перед вами появится главное окно VLC (рис.1). Теперь можно просмотреть локальный видеофайл, создать поток или подключиться к уже имеющемуся. Разберем со всем этим по порядку.

Чтобы просмотреть файл, введите в командной строке:

```
$ vlc -vvv my_videofile.mpg
```

Программа сама подберет необходимый декодер. В случае неудачи его можно задать вручную параметром --codec.

```
$ vlc -vvv --codec ffmpeg my_videofile.mpg
```

Просмотр VCD или DVD начинается командой

```
$  vlc -vvv vcd:/dev/cdrom:@1:1
```

То же самое можно сделать и через меню. Просто откройте пункт Файл и выберите источник сигнала, файл, папку или диск.

Создание и просмотр потока

VLC (как и *VLS*) может создавать два типа видеопотоков: unicast и multicast (возможно применение обеих вариантов одновременно). В первом случае видеофайл разбивается на пакеты и отправляются по индивидуальному IP-адресу, указанному программе при запуске. Принимающая сторона просто считывает данные, поступающие на открытый для приема UDP-порт (по умолчанию используется 1234). Во втором случае сервер организует передачу на специальный групповой IP-адрес, с которого и считывают информацию многочисленные клиенты. Для создания unicast-потока следует ввести команду вроде:

```
# vlc -vvv videofile.avi --sout udp://192.168.0.42 --ttl 1
```

где videofile.avi – транслируемый видофайл, udp://192.168.0.42 – unicast IP-адрес (имя DNS) отдельного компьютера, ttl означает Time To Live т.е. время жизни пакета. Каждый маршрутизатор уменьшает значение TTL на единицу и, когда оно становится равным нулю, пакет уничтожается. Таким образом, установив значение TTL в 1, мы не допустим передачи трансляции в глобальную сеть.

Можно отправить в сеть и содержимое DVD диска.

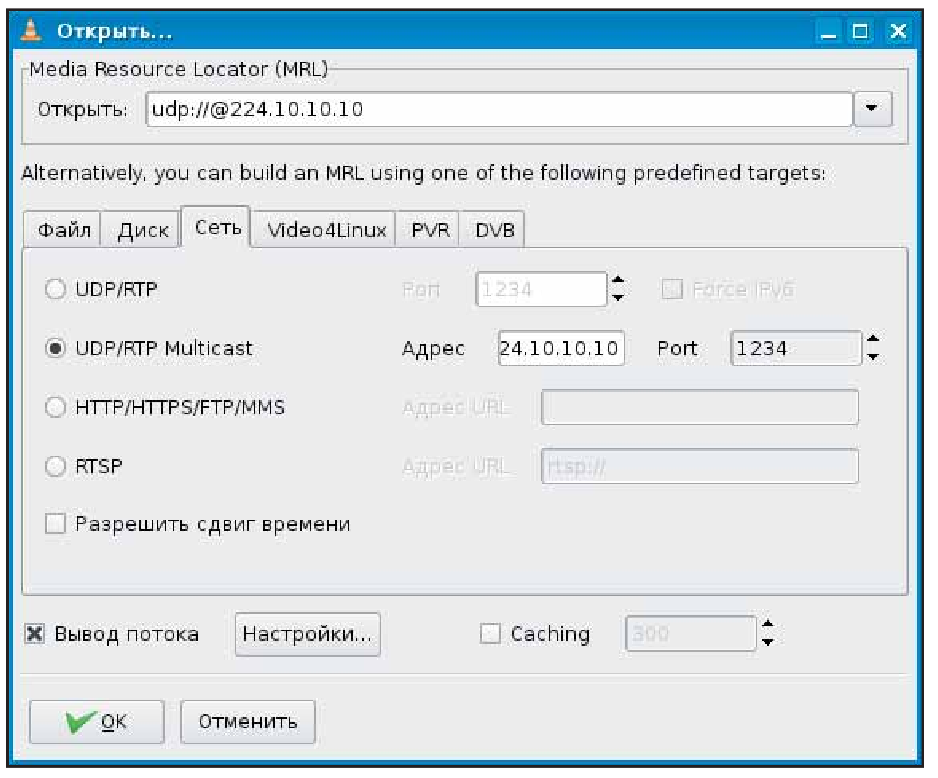
```
# vlc -vvv dvd:/dev/dvd --sout udp://example.org
```

Для трансляции DVD или VCD под Unix потребуется право на

› **Рис. 2**

Вывод потока
Stream output MRL
Destination Target: ,dst=std{access=udp,mux=ts,url=224.10.10.10:1234,sap}} :sout-all
Output methods
Воспроизводить локально
Файл — Имя файла — Обзор... — Dump raw input
HTTP — Адрес — Port 1234
MMSH — Адрес — Port 1234
RTP — Адрес — Port 1234
UDP — Адрес 224.10.10.10 — Port 1234
Encapsulation Method
MPEG TS, MPEG PS, MPEG 1, Ogg, ASF, MP4, MOV, WAV, Raw
Transcoding options
Видео кодек mp4v — Bitrate (kb/s) 1024 — Scale 1
Аудио кодек mpga — Bitrate (kb/s) 192 — Каналы 2
Кодек субтитров dvbs — Subtitles overlay
Miscellaneous options
SAP announce — Group name — Название канала
Select all elementary streams — Time-To-Live (TTL) 1
OK — Отменить

»



› Рис. 3

Streaming/Transcoding Мастер — Выберите вход

› Рис. 5

» запись в **/dev/dvd** (**/dev/cdrom**). Обычно для этого необходимо включить пользователя, от имени которого запускается сервер, в группу cdrom. Более грубый, но и более простой подход состоит в передаче этого права всем и каждому:

```
# chmod 666 /dev/dvd
```

Аналогичным образом можно транслировать сигнал и с других источников.

Подключиться к unicast-потоку можно следующим образом.

```
# vlc -vvv udp:
```

В случае использования порта, отличного от 1234, необходимо явно указать его номер:

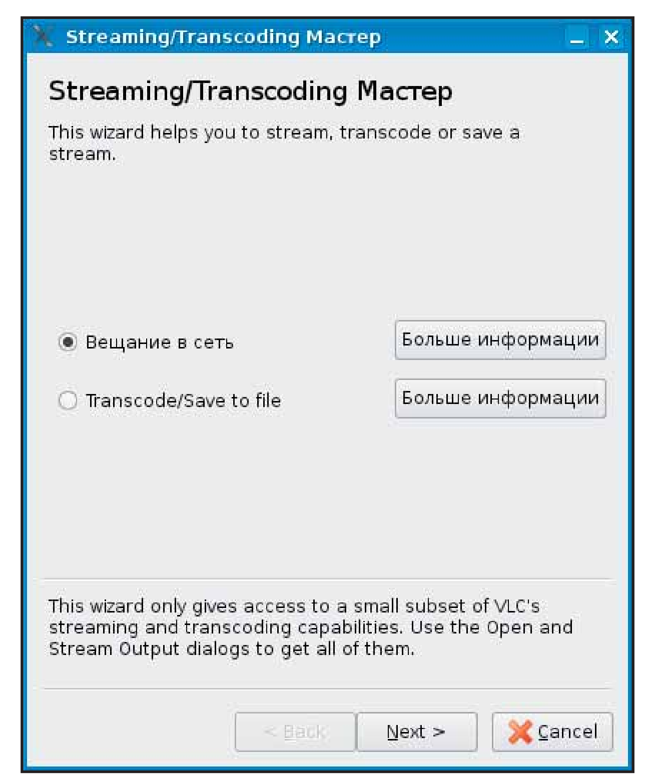
```
# vlc -vvv udp:@:9876
```

Забегая немного вперед, скажем, что при подключении к multicast-потоку необходимо добавить еще и IP-адрес.

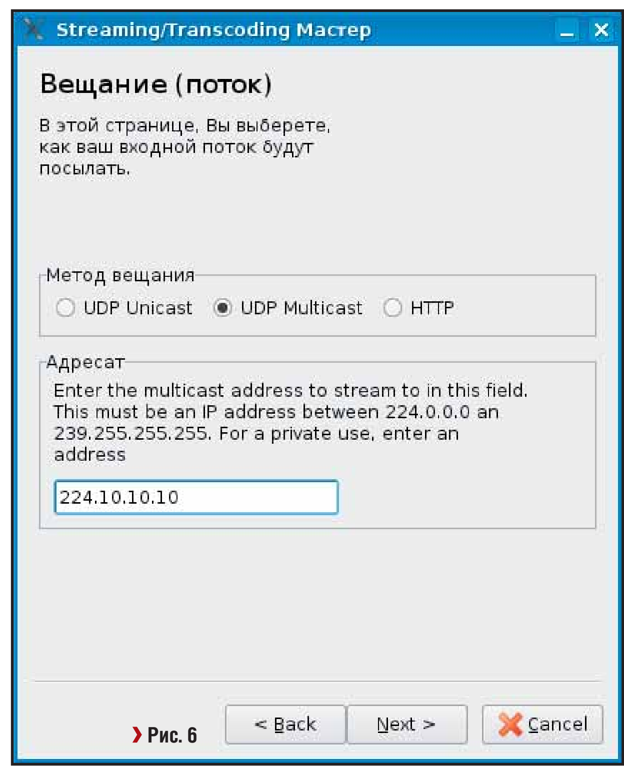
```
# vlc -vvv udp:@multicast_address[:server_port]
```

Для широковещательных передач зарезервирован специальный диапазон IP-адресов: от 224.0.0.0 до 239.255.255.255. Для организации multicast-потока можно выбрать любой понравившийся адрес из этого диапазона и настроить его использование как на сервере, так и на клиенте. Для трансляции можно воспользоваться и стандартным для любой сети широковещательным адресом, как правило, заканчивающимся на 255, но в большой сети это может помешать нормальной работе некоторых служб.

VideoLAN поддерживает и так называемое HTTP-вещание, когда



› Рис. 4



› Рис. 6

клиенты подключаются к почти обычному Web-серверу, только в ответ на запрос получают не текстовый файл, а видеоданные. Сервер в таком случае запускается так.

```
# vlc -vvv input_stream --sout '#standard{access=http,mux=ogg,url=server.example.org:1234}'
```

Клиенту (*VLC* или другой программе, например, *Xine*) достаточно указать стандартный URL, начинающийся с http://:

```
# vlc http://server.example.org:1234
```

Поток можно создать и с помощью меню. Для этого выберите источник (Файл – Открыть файл). В появившемся диалоговом окне отметьте нужный файл и поставьте флажок напротив Вывод потока (Stream Output), затем нажмите Настройка. В открывшемся окне (рис.2) можно либо перечислить параметры командной строки в поле Destination Target, либо (что более удобно), воспользоваться группой элементов Output Methods.

Например, для организации потока UDP multicast необходимо поставить флажок напротив пункта UDP. Параметры Адрес (224.10.10.10) и порт (1234) можно оставить без изменения Активировав пункт Воспроизводить локально, вы сможете контролировать, что уходит в сеть, на компьютере-сервере. Для приема потока на другом компьютере используйте меню Файл – Open Network Stream, в появившемся окне укажите реквизиты multicast-сервера. Полученный поток можно тут же ретранслировать на другой адрес, как показано на рис.3.

Существует и третий вариант – использование мастер-трансляции и перекодирования, которое вызывается из одноименного пункта меню Файл. В этом случае вам необходимо будет пройти всего пять шагов:

– выбор режима работы – Вещание в сеть (рис.4)

– выбор источника сигнала – поток или список (рис.5)

– выбор варианта выходного потока – UDP unicast, multicast или HTTP (рис.6)

– выбор формата форматирования пакета (рис.7)

– установки TTL и анонсов SAP (рис.8)

После нажатия на кнопку Finish можно подключаться к потоку и проверять результат. Как можно видеть, при использовании мастера некоторые опции оказываются недоступными.

Продвинутые возможности

На закуску поговорим о некоторых возможностях VLC, выходящих за пределы базовой функциональности «потокового вещателя». Например, очень полезная в хозяйстве вещь – transcoding. При этом исходный файл перекодируется в любой из поддерживаемых VLC форматов «на лету» и затем уже отправляется в странствие или сохраняется на жестком диске (опция File во вкладке Stream Output). Активировав пункты Audio и Video codec в поле Transcoding Options, можно установить кодек, битрейт и количество аудиоканалов. В командной строке все эти функции можно подключить через опцию transcode, если, конечно, вам нравятся конструкции вида:

```
# vlc -vvv dvd:/dev/dvd –sout '#transcode{vcodec= DIV3, acodec=vorb,vb=800,ab=128,channels=2,deinterlace}:standard{access=udp,mux=ts,url=239.255.12.42,sap=TestStream}'
```

Подобным образом можно легко собрать очень неплохую и к тому же универсальную программу «ограбления» видеодиска и захвата видео. С помощью модуля duplicate есть возможность разделить входящий поток на два и затем каждый из них обработать по своему усмотрению.

```
# vlc -vvv videofile.avi --sout '#duplicate{dst=display,
dst="transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128}:
duplicate{dst=standard{access=udp,mux=ts,url=192.168.1.2},
dst=standard{access=udp,mux=ts,url=192.168.1.12}"}'
```

Все параметры, чтобы не вводить их каждый раз, можно прописать в конфигурационном файле. По умолчанию используется **$HOME/.vlc/vlcrc**.

В итоге, после небольших манипуляций, мы получили свой сервер, транслирующий видеопоток. Ответы на все вопросы о возможностях программ проекта VideoLAN можно найти в документации, которой более чем предостаточно на официальном сайте. LXF

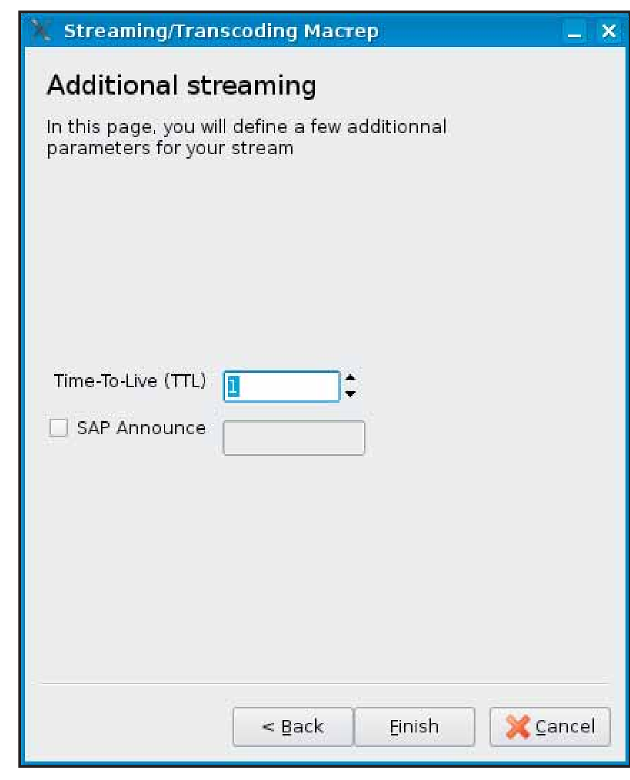


Рис. 8

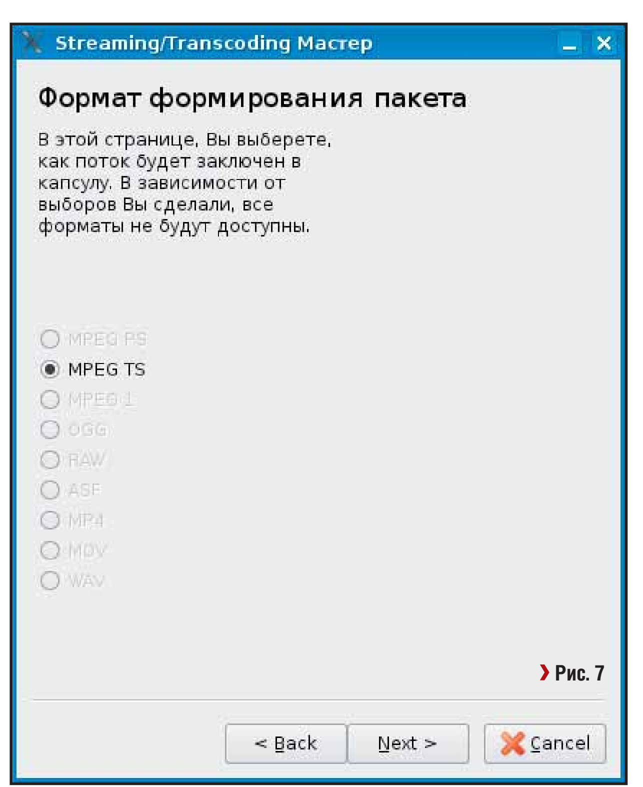


Рис. 7





Ответы

Есть вопрос по Open Source? Пишите нам по адресу: answers@linuxformat.ru

В этом месяце мы отвечаем на вопросы по:

1 Smkiso
2 USB
3 Webmail
4 Составлению отчетов
5 Grub
6 Принтерам Canon
7 Mon
8 Fam
9 KFind
10 Оформлению GTK-программ
11 Шрифтам
12 VNC
13 Модемам
★ Мониторингу сетевого трафика
★ Настройке интернет-подключения

1 SUSE: из DVD в CD

В SUSE из августовского номера (**LXF82**) дала мне отличную возможность установить Linux и начать его изучение. У меня есть компьютер с DVD-приводом, но я хочу попробовать Linux не на нем, и я не представляю, как создать комплект инсталляционных CD для другого компьютера.

Алан Ханимен [Alan Honeyman]

О Тот диск включал скрипт *mkiso*, предназначенный для создания образов CD из DVD. Однако он работает только в Linux. В этом месяце мы выложили на диск скрипт, позволяющий сделать то же самое в Windows. Эта несложная процедура займет немного больше времени, чем обычно, поскольку скрипт расположен не на том же самом диске. Начните с копирования **winmkiso.bat** с DVD куда-нибудь на жесткий диск. Неважно куда, по окончании процедуры его можно будет удалить. К примеру, Вы скопировали его на диск **C:**, а диск **D:** – это Ваш DVD-привод. Откройте командную строку и напечатайте

```
D:
cd distros\suse
c:\winmkiso
```

Вам придется довольно долго ждать, потому что для создания каждого образа CD скрипт просматривает все содержимое DVD-диска. Так уж устроена *Jidgo* – программа, выполняющая черную работу. Linux-версия кэширует данные при первом просмотре, чтобы увеличить скорость следующих; Windows-версия, по идее, делает то же самое, но в реальности создание каждого образа весьма продолжительно. В конце концов Вы получите пять образов на **C:**. Можете записать их с помощью *Nero* или другой программы прожига CD, но обязательно через опцию записи ISO-образа.

Если Вы захотите поместить создаваемые образы в другое место, укажите в команде *winmkiso* путь. Например, если Ваш DVD-привод **E:** и Вы хотите сохранить образы в **D:\suse**, выполните:

```
E:
cd distros\suse
c:\winmkiso D:\suse
```

Убедитесь, что указанный путь существует, иначе *winmkiso* просто будет расходовать ваше время, а в конце концов скажет, что не может сохранить ISO-образ. **НБ**



› *Winmkiso* поможет создать комплект CD-дисков SUSE из DVD.

2 Верь мне

В Мне подарили цифровую фотокамеру (производства Trust), но я не сумел заставить *DigiKam* распознать ее. Тем не менее, судя по **/var/log/messages**, новое USB-устройство было обнаружено, а камера включилась. Я ничего не знаю про USB, но я знаю, что камеры нет в **/etc/fstab**. Может, проблема в этом? И вправду ли *DigiKam* – лучшая KDE-программа для этих целей?

chris_debian, с форума LXF

О Вам не нужна запись в **fstab** для того, чтобы работало автомонтирование в KDE: без нее оно работает даже лучше. Однако не все цифровые камеры работают как стандартные USB-накопители; некоторые используют специфичные протоколы. Показано ли в **/var/log/messages**, что система нашла разделы на USB-устройстве? Строчки вроде этих помогут Вам убедиться, что камера подключается как стандартный USB-накопитель:

»

Наши эксперты

» Мы найдем эксперта на любой вопрос! Вы получите ответ на все: от проблем с установкой или модемом до сетевого администрирования; главное – спросить!



Нейл Ботвик
Владелец ISP и экс-редактор дисков для нашего журнала, Нейл считает, что в Linux он от скуки на все руки.



Майк Сондерс
Майк был одним из создателей прототипа LXF – Linux Answers. Его специальности – программирование, оконные менеджеры, скрипты инициализации и SNES.



Дэниель Кингшот
Эксперт по вопросам системных администраторов, Дэниель возглавляет службу интенсивной технической поддержки Rackspace, управляет более чем 40 пользователями и имеет значительный опыт работы в Linux

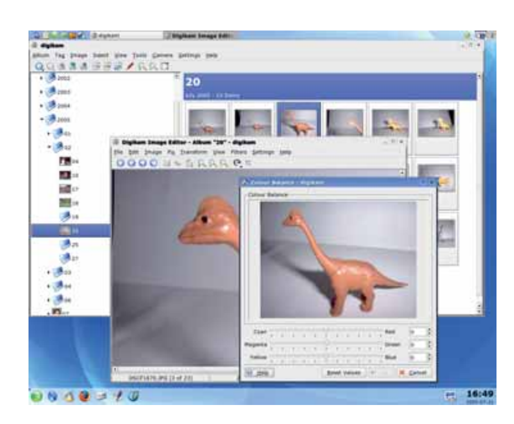


Ник Вейч
В свободное от исчеркивания текстов красными чернилами время Ник возится с Linux-графикой и 3D-приложениями; он у нас отвечает за простые вопросы!



Валентин Синицын
В свободное от работы время редактор нашего журнала разрабатывает *KNetworkManager* и другие открытые приложения. Он с радостью поможет вам в вопросах использования Linux на рабочем столе.

КУДА ПОСЫЛАТЬ ВОПРОСЫ:
Пишите на м по адресу: answers@linuxformat.ru или спрашивайте на форуме: www.linuxforum.ru



› *DigiKam* умело обращается с фотографиями.

»

```
usb-storage: waiting for device to settle before scanning
Vendor: NIKON Model: NIKON DSC E3200
Rev: 1.00
Type: Direct-Access                ANSI SCSI revision: 02
SCSI device sda: 2012160 512-byte hdwr sectors (1030 MB)
...
SCSI device sda: 2012160 512-byte hdwr sectors (1030 MB)
sda: sda1
```

Если подобных строк нет, значит, камера не является стандартным накопителем. Однако *DigiKam* все-таки должен распознать ее, если она поддерживается *Gphoto2* и *Gphoto2* у Вас есть. *Gphoto2* – клиент для библиотеки *libGphoto2*, используемой *DigiKam*. Если он не установлен, пакет можно найти на дисках дистрибутива. Вы должны запустить его от обычного пользователя и от root – любое различие в выводе укажет на проблему с правами доступа.

Чтобы узнать, поддерживается ли Ваша камера, запустите

```
gphoto2 --auto-detect
```

и посмотрите на вывод. Если камера не обнаружилась, проверьте архивы списков рассылки **www.gphoto.org** и пошлите разработчикам данные Вашей камеры.

DigiKam отлично управляется с цифровыми фотографиями, но если Вам нужно только скопировать изображения с фотоаппарата, это можно сделать через *Konqueror*, набрав в адресной строке **camera:/**. Отобразятся все подключенные и обнаруженные *libGphoto2* камеры. **НВ**

3 E-mail отовсюду

В У меня почтовый сервер на Linux (*SquirrelMail*). Как я могу проверить почту с любой Windows-машины без установки дополнительного ПО?

Энди Уайетт (Andy Wyatt)

О Вы можете читать почту с помощью любого стандартного почтового клиента, даже *Outlook Express*. Если Вы не хотите настраивать для этого почтовый клиент (например, в случае временного пользования чужим ПК), лучше всего установить на сервере web-интерфейс: это позволит читать почту откуда угодно при помощи только web-браузера.

SquirrelMail (**www.SquirrelMail.org**) – один из наиболее популярных и долгоживущих серверов web-почты. Вам нужно запустить сервер IMAP, поскольку большинство web-интерфейсов работают с ним. *SquirrelMail* – приложение PHP, запускаемое через web-сервер, так что понадобится установленный и настроенный Apache (или другой web-сервер). Установленный и настроенный (это хорошо документированный процесс) web-интерфейс даст Вам доступ к Вашему почтовому ящику через практически любой браузер.

Кроме *SquirrelMail*, есть и другие решения. Лично мне нравится *RoundCube* (**www.RoundCube.net**). Это Ajax-проект, и хотя его версия 0.1beta2, он выглядит вполне стабильным и функциональным. Выберете Вы какое-то решение из этих двух или предпочтете *NeoMail* (**http://neocodesolutions.com/software/neomail**), зависит от Ваших потребностей. Если доступ через web-интерфейс Вам нужен нечасто, подойдут все три проекта (я все же порекомендовал бы *RoundCube*). Если Вас ждет интенсивная работа и могут потребоваться какие-то продвинутые функции полноценного почтового клиента, протестируйте все три проекта и выберите наиболее подходящий.

Можно установить их все, поместив каждый в отдельную директорию на сервере, и определить, какой из них Вам удобнее. **ДК**

4 Потайной раздел

В Мой тесть недавно заинтересовался GNU/Linux, и я посоветовал ему скачать Ubuntu. Я надеялся, что установка пройдет гладко, однако он столкнулся с проблемой, которая поставила меня в тупик. Может быть, решение и простое, но не для меня.

У него новый P4 Dell, поставляемый с предустановленной Windows XP. Есть 10 ГБ свободного места для установки. Обычная процедура установки прошла гладко и попросила перезагрузки. После перезагрузки *Grub* выдал ошибку 21. Покопавшись, тесть обнаружил, что Dell помещает на диск небольшой скрытый раздел с фирменными утилитами Dell. Видимо, MBR [Master Boot Record] на этих машинах куда-то смещен. Опрос на форумах Ubuntu показал, что это и вправду проблема и что никто не знает хорошего решения.

Так как настроить *Grub*, чтобы он нормально запускался? У меня нет физического доступа к машине, однако я знаю, что на ней только один жесткий диск. После установки список разделов выглядит примерно так:

» **hda1** Суперсекретные файлы Dell.
» **hda2** Windows.
» **hda3** Загрузочный раздел.
» **/ swap**.

Майкл Маркс [Michael Marks]

О Вы не сказали, что за модель Dell у Вашего тестя, но обычно утилиты Dell расположены на **hda1**, а загрузочным разделом с Windows является **hda2**. MBR должен быть на обычном месте, иначе BIOS не нашла бы разделы. Ошибка 21 – это ошибка второго этапа загрузки. То есть *Grub* уже загрузился из MBR и нашел в **/boot** файлы второго этапа.

Ошибка 21 означает «Выбранного диска не существует», так что скорее всего неправильно настроен *Grub*: возможно, он пытается загрузить ядро с несуществующего раздела. Если *Grub* способен загрузить Windows, значит, так оно и есть (и это доказывает, что сам *Grub* в порядке).

»



Краткая справка...

Автодополнение

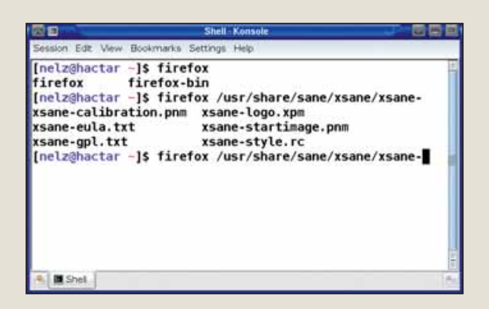
Применим все, что делает кодирование эффективнее!

Есть ряд причин, по которым пользователи Linux избегают использования командной строки. Одна из них – забывчивость. Еще есть чувство, что выбрать файл мышкой гораздо быстрее, чем печатать полный путь к нему. А если вы плохо набираете, всегда есть шанс, что команда выполнится неверно из-за неправильного ввода.

Благодаря самой дружественной к пользователю функции командных оболочек, автодополнению по клавише Tab, эти причины уже не должны нас тревожить. Что это такое? Проще всего показать на примере. Пусть вам надо прочитать файл **/usr/share/sane/xsane/doc/sane-xsane-fax-doc.html** (такой есть у меня на компьютере). Можно запустить браузер и попытаться без ошибок набрать путь к нему. Или воспользоваться командной строкой и ввести

```
fire[TAB]/us[TAB]sha[TAB]sa[TAB]x[TAB]d[TAB]sa[TAB]x[TAB]f[TAB]
```

Первый Tab будет искать команду (*Firefox*) в путях поиска (PATH). Следующие будут пытаться дополнить вводимый путь. Это не только быстрее, чем полностью набирать путь, но и гораздо безопаснее по части ошибок, поскольку выбираться будут только существующие пути.



› Автодополнение делает ввод длинных команд и путей более быстрым и аккуратным.

А что если обнаружится несколько совпадений? Тогда оболочка дополнит столько символов, сколько сможет, и если вы еще раз нажмете Tab, выдаст список возможных вариантов. Вы можете ввести еще один-два символа и снова нажать Tab. Этим же способом можно узнать, какие команды доступны.

При правильном использовании автодополнение значительно ускорит работу в командной строке.

» Нажмите Esc для входа в меню *Grub*, подсветите пункт Linux и нажмите e. Вы должны увидеть нечто вроде этого:

```
root (hd0,0)
kernel /boot/vmlinuz-2.6.15-23-386 root=...
```

Велика вероятность того, что значение root неверно. Нажмите c и в подсказке *Grub* наберите

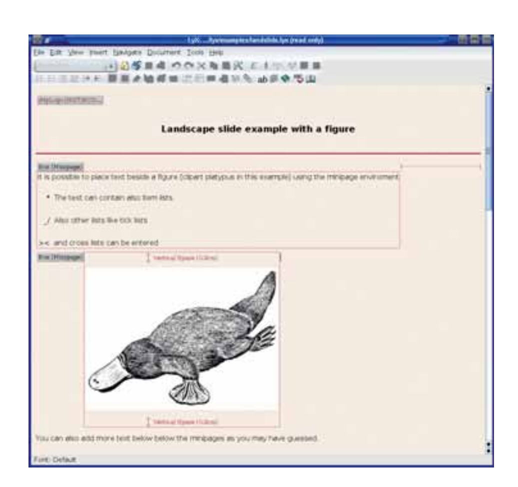
```
find /boot/vmlinuz-2.6.15-23-386
```

используя имя ядра, показанное в предыдущей строке. Это вернет местоположение раздела, содержащего образ ядра: возможно, это будет (**hd0,2**) или (**hd0,4**), в зависимости от того, расположен ли **/boot** на первичном или логическом разделе. Нажмите Esc для возврата в меню, подсветите строчку с 'root', нажмите e и измените ее согласно результату команды *find*. Затем нажмите Enter для применения изменений и 'b' для загрузки.

Убедившись, что все работает, можете изменить конфигурационный файл, запустив в терминале

```
sudo nano /boot/grub/menu.lst
```

Настройки меню вы найдете под строкой *## ## End Default Options ##*. Плохо лишь то, что для этого потребуется физический доступ к компьютеру, или уж объясните все своему тестю. **НБ**



› ***Lyx*** **– мощная программа типографского набора, идеальна для высококачественных отчетов.**

5 Автоверстка

В Хотел бы напечатать картинку в BMP или JPEG и текст в одном отчете. Есть ли бесплатная утилита, позволяющая сделать это из командной строки? Хорошо бы генерировать отчеты скриптом.

et_phonehome_2, с форума LXF

О Да, есть даже несколько способов. Выбор зависит от качества вывода и времени, которое Вы готовы на это потратить. Самый простой вариант – записывать отчет в HTML, тогда его можно просмотреть или напечатать в любом web-браузере. Следующий скрипт берет имена файлов с изображением и текстом и пишет HTML в стандартный вывод. Это очень простой пример, но Вы поймете идею:

```
#!/bin/sh
echo "<html><head><title>Мой отчет</title></
head><body>"
echo "<img src=\"$1\" align=\"right\">"
cat $2
echo "</body></html>"
```

На другом конце спектра находятся пакеты на базе *TeX*, типа *teTeX* или *LyX*. Эти программы типографского набора дают полный контроль над оформлением документа. Изучить их непросто, но результат оправдает Ваши ожидания. Исходные файлы *TeX* – обычный текст, и его легко генерировать из командной строки с использованием шаблона и небольшого скрипта.

LyX – более простое средство для создания файлов *TeX*. Это графическое приложение, но однажды создав в нем шаблон, Вы можете манипулировать им из скрипта. Можно разбить шаблон на три куска и сделать что-то вроде этого:

```
cat template1.lyx > report.lyx
echo /путь/к/изображению >> report.lyx
cat template2.lyx текст.txt template3.lyx >>report.lyx
```

Или вариант посложнее: используйте sed для подстановки путей к файлам изображения и текста в шаблоне. Полученный файл **report.lyx** можно преобразовать в несколько высококачественных форматов. Например,

```
lyx --export pdf report.lyx
```

сгенерирует отчет в формате PDF. *LyX* – мощная программа с хорошей документацией. Обязательно попробуйте ее.

Как альтернативу можно рассмотреть использование скриптов в программах вроде *OpenOffice.org* (ознакомьтесь с нашим учебником в LXF80–LXF84), *AbiWord* или *Scribus*. **НБ**

6 Упрямый Canon

В У меня компьютер с Windows XP и SUSE 10.1 и лазерный принтер Canon LBP-1120. Под Windows принтер работает отлично. Беда в том, что как я ни бьюсь, с настройкой его в Linux ничего не получается. Я скачивал и устанавливал CAPT-драйверы, настраивал принтер, и ничего не происходило.

Единственное, чего удалось добиться – принтер прогнал чистый лист бумаги. Все, что я пытаюсь напечатать, просто сидит в очереди, и приходится скидывать документы себе на почту, принимать ее в Windows и печатать их оттуда.

Найджел Норфолк [Nigel Norfolk]

О Это Win-принтер – один из тех, чей драйвер выполняет ряд функций прошивки. Как и в случае с родственными им win-модемами, »

Часто задаваемые вопросы

Широкополосное соединение

Почему USB-модемы – это зло.

» Что вообще такое – широкополосное соединение?

Исходное определение – это сетевое соединение, позволяющее принимать сетевой телевизионный сигнал в реальном времени, то есть не менее 2 Мбит/сек. В наше время любое соединение быстрее модемного считается широкополосным. Большинство провайдеров предлагают скорости от 128 Кбит/сек до 8 Мбит/сек.

» Сколько бывает типов широкополосного соединения?

Есть два основных типа. ADSL использует стандартные телефонные линии, но передает данные на гораздо более высоких частотах, чем голосовые звонки. Кабельные модемы используют для передачи трафика сети кабельного ТВ. Оба типа могут работать через отдельный модем либо через телеприставку. Отдельный модем дает большую гибкость в размещении ПК: не обязательно ставить его рядом с телевизором.

» Все ADSL-провайдеры в стране пользуются теми же телефонными сетями. Так имеет ли смысл выбирать?

Имеет. Телефонная линия лишь обеспечивает канал между модемом и провайдером. А вот с Интернетом некоторые провайдеры работают лучше, некоторые хуже; у кого-то есть ограничения по трафику или портам; кто-то предоставляет статические адреса, и можно установить собственный сервер…

» Какой брать модем?

В случае с кабельными сетями, вам придется использовать тот, что установил ваш провайдер. Может быть, у вас будет выбор, подключить ли его к USB или Ethernet. Всегда выбирайте Ethernet – он разработан для сетей, USB – нет.

Некоторые ADSL-провайдеры предоставляют модем «бесплатно». Это USB-модем, и его лучше оставить в коробке. Ethernet-вариант стоит не дороже 35 евро. Для него не нужны драйверы; просто подключите его, забейте нужный адрес в браузер и настройте параметры вашего провайдера.

» А маршрутизатор мне нужен?

Не знаю – это зависит от количества компьютеров у вас дома. Роутер позволит вам использовать одно соединение на множестве компьютеров. Большинство Ethernet ADSL-модемов включают маршрутизатор, но для кабельного модема придется покупать его отдельно. Некоторые маршрутизаторы имеют точки беспроводного доступа, это позволит вам выходить в сеть через Wi-Fi с ноутбука или КПК.



› **Ethernet-модем или маршрутизатор легко настроить в web-браузере.**

» заставить их работать вне Windows – не самая тривиальная операция, и не факт, что она завершится успешно. У Вас есть выбор: существует официальный драйвер от Canon – его-то, я думаю, Вы и пробовали, и еще один, рекомендованный www.linuxprinting.org и доступный на сайте www.boichat.ch/nicolas/capt. Я бы посоветовал попробовать оба драйвера, а также последовать инструкциям с http://linuxprinting.org/show_printer.cgi?recnum=Canon-LBP-1120.

Диагностируя проблемы печати, в первую очередь Вы должны проверять журналы CUPS. Наберите в терминале

```
tail -f /var/log/cups/error_log
```

и попытайтесь напечатать страницу. На терминале появятся сообщения, записываемые в лог-файл, они часто дают ключ к решению. По умолчанию сообщения довольно скудны. Если Вам нужна детальная информация, отредактируйте от лица root файл **/etc/cups/cupsd.conf**, найдите в нем строку

```
LogLevel info
```

и замените info на debug. Перезапустите CUPS из YaST или терминала:

```
/etc/init.d/cups restart
```

Теперь error_log будет содержать больше данных об ошибках. В качестве руководства перед приобретением принтера каждый линуксоид должен использовать сайт www.linuxprinting.org. **НБ**

7 Мониторинг сервисов

В На моем сервере работает ряд сервисов. Есть ли способ следить за ними и перезапускать их, когда они умирают? Я подумывал о заданиях по типу *Cron*.

Генри Робертс [Henry Roberts]

О Есть несколько программ, написанных специально для этой задачи, и самая популярная, вероятно, *Mon*, доступный на www.kernel.org/software/mon. У этой утилиты довольно длинный список зависимостей (в основном модулей Perl), поэтому лучше всего установить ее через менеджер пакетов вашего дистрибутива. *Mon* можно установить и на тот компьютер, за сервисами которого вы намерены следить, и на удаленный – доступный через сеть. Последний вариант лучше, поскольку так вы сможете узнать, что ваш сервер полностью упал.

Mon настраивается с помощью файла конфигурации **/etc/mon**. Вот, например, его часть, отвечающая за мониторинг web-сервера:

```
hostgroup servers www.example.com
watch servers
 service http
  interval 5m
  monitor http.monitor
  period wd {Sun-Sat}
   alertevery 1h
   alert mail.alert webmaster@example.com
```

Утилита будет соединяться с сервером каждые пять минут и отсылать письмо с уведомлением, если он окажется недоступным. Параметр alertevery указывает, что при недоступности сервера программа должна продолжать проверки, но отправлять уведомления не каждые пять минут, а с часовым интервалом. *Mon* умеет не только наблюдать за сервисами, но и следит за дисковым пространством и процессами, помогая предотвратить атаки типа отказ в обслуживании. В *Mon* предусмотрены разные виды уведомлений, включая отсылку сообщения на пейджер (а то зачем отправлять письмо, если почтовый сервер рухнул?). Модули слежения и уведомления – обычные скрипты на Perl, их можно подправить и приспособить под свои нужды. На сайте *Mon* есть множество пользовательских вариантов.

Другая похожая программа – *Monit* (www.tildeslash.com/Monit). Она работает по тому же принципу, что и *Mon*, но спроектирована специально для запуска на целевом сервере и способна выполнять самостоятельные операции по «спасению», а не просто уведомлять администратора. *Monit* умеет перезапускать сервисы, а кроме того, содержит встроенный web-сервер, дающий возможность просмотреть статистику по сервисам с удаленного компьютера. Самый безопасный подход – запускать *Mon* удаленно, а *Monit* локально. **ДК**

8 Инспектор трафика

В Мне нужно следить, как используют канал мои серверы. Как можно вести учет сетевого трафика для всех или каких-то определенных интерфейсов?

Том Райс [Tom Rice]

О Есть множество программ, отображающих информацию о трафике на каждом интерфейсе, большинство из них берут данные из **/proc**. А отличаются они обычно способом представления полученной статистики.

Для простого обзора хороший выбор – *Vnstat*. Он доступен на http://humdi.net/Vnstat и, возможно, в репозитарии Вашего дистрибутива. *Vnstat* обычно запускается раз в час через *Cron*, собирает статистику из **/proc** и добавляет ее в базу данных. Вы можете обратиться к этой базе, запустив *Vnstat* из командной строки. Есть опции для отображения статистики за день, неделю или месяц, а также множество вариантов настройки вывода.

Если Вам нужно нечто покруче простого отчета в ASCII, попробуйте *Traffic-vis* (www.mindrot.org/traffic-vis.html). Этот пакет содержит несколько утилит; основную работу выполняет *Traffic-collector*, запущенный постоянно. *Traffic-collector* отслеживает трафик на определенных сетевых интерфейсах и сохраняет данные в файле. Этот файл предназначен не для чтения напрямую, а для передачи другим программам, которые преобразуют его в отчет формата HTML, PostScript, обычного текста или GIF. Опция преобразования в HTML будет интересна, если Вы хотите следить за web-сервером. Тогда из CGI-скрипта можно запустить *Traffic-tohtml* и получать самую свежую информацию прямо в браузере. Есть и другие утилиты для обработки данных о трафике. Например, *Traffic-exclude* будет полезна, если у Вас есть ограничения по трафику или трафик платный, и Вы хотите знать, много ли байт уходит за пределы локальной сети, не вникая во внутреннее прохождение информации.

9 Назойливый KFind

В Я использую Kubuntu 6.06 с пакетами Ichthux, но мой вопрос скорее касается любого дистрибутива с KDE. Стандартная утилита поиска файлов сканирует каждую директорию, начиная с корневой и включая поддиректории в **/mnt**. Это значит, что *KFind* ищет файлы в других моих системах и, соответственно, зря тратит время. Есть ли возможность исключить из поиска **/mnt**?

Все, что мне удалось придумать – размонтировать **/mnt** перед каждым использованием *KFind*, но это создает проблемы.

Дэйв [Dave], с форума LXF

О *KFind* – это просто графический интерфейс к двум стандартным утилитам: *locate* и *find*. К сожалению, он не дает доступа ко всем опциям *find*, в частности, к установке областей поиска. Все, что можно сделать – указать начало поиска. Это не страшно, если надо найти файл в домашней директории (выбор по умолчанию), однако при просмотре всей файловой системы возникают проблемы вроде вашей.



› Перед покупкой принтера узнайте на www.linuxprinting.org, хорошо ли он поддерживается в Linux.



Вопрос-победитель (английская версия)

В ожидании файлов

В У меня проблемы с обновленным Red Hat-сервером на работе. У нас есть апплет, устанавливающий FTP-соединение с сервером, чтобы пользователи закачивали файлы. Он работает нормально.

Проблема в скрипте, который запускается на сервере для просмотра даты изменения директории, куда закачиваются файлы. Когда дата меняется, он обрабатывает новый файл. Это работало на старом сервере, но на новом дата меняется еще до полной закачки файла, инициализирует процесс, и мы получаем неполные файлы.

Есть ли способ установить параметры обновления даты модификации для директорий? Дело в системе или в FTP? Сейчас у нас RHEL ES 4, раньше был RHEL ES 2.

Kevsan, с форума LXF

О Проблема в том, что директория меняется дважды: при открытии нового файла в начале закачки и при его закрытии по завершении. Недавно я настраивал нечто подобное и нашел выход в использовании сервиса *Fam* (File Alteration *Monitor* – монитор изменения файлов), способного различать эти события. Вам нужно установить *Fam* и убедиться, что при загрузке запускается **famd**. После этого нужна программа, чтобы следить за изменениями файлов в директории и информировать об этом. Я нашел утилиту *fileschanged* идеальной для запуска из скриптов. Вы можете взять *fileschanged* с http://fileschanged.sourceforge.net и запустить ее так:

```
fileschanged --show changed --exec /usr/local/bin/ourscript /var/ftp/somedir/
```

Опция --show велит *fileschanged* следить только за изменениями в директории, опуская сообщение о появлении файла (*fileschanged* также отмечает удаление или исполнение файлов). Когда утилита получает уведомление, она запускает скрипт с двумя аргументами. Первый – буква, отображающая тип изменения (M – модификация файла). Второй аргумент – имя файла. С этой информацией ваш скрипт будет знать имя закачанного файла. Вы также можете найти полезной опцию --timeout для увеличения задержки уведомления. **ДК**

К счастью, с помощью флажка Использовать файловый индекс (Use files index), вы можете выбрать *locate* вместо *find*. *Locate* обращается к базе данных, построенной командой updatedb для ускорения поиска, однако находит только файлы, существовавшие до последнего обновления базы. Updatedb обычно запускается ежедневно или еженедельно как задача *Cron*. Пути поиска *locate* настраиваемы, так что Вы можете добавить **/mnt** в список PRUNEPATHS в файле **/etc/updatedb.conf**.

Для достижения желаемой гибкости, стоит изучить сами команды *find* и *locate*. Например, строка

```
find / /home -xdev -iname `*.pdf'
```

будет искать все PDF-файлы в каталогах **/** и **/home**, игнорируя (благодаря опции **-xdev**) другие файловые системы типа **/proc**, **/dev** или смонтированные под **/mnt** или **/media**. man-страницы *locate* и *find* содержат много полезной информации, но главное, что следует запомнить – *locate* быстра, зато *find* гораздо гибче, поскольку может учесть имя, тип и возраст файла, а также и директории. Размонтирование файловых систем из **/mnt**, конечно, поможет, но вызовет проблемы, если у Вас открыт файл на какой-то из этих систем. В любом случае, без него можно обойтись. **НБ**

10 Увеличение шрифтов в GTK

В Ищу информацию о том, как увеличить размер шрифтов в *Xara Xtreme*, *Gimp* и подобных программах, на которые не влияют настройки System Config > Appearance и Themes > Fonts.

Я использую SimplyMEPIS с диска LXF84 – и доволен.

Билл Эпплби [Bill Applebee]

О *Xara Xtreme* и *Gimp* используют виджеты *GTK2*, а System Config влияет только на KDE-приложения. Я бы посоветовал Вам установить пакет *gtk2-engines-gtk-qt*, он дает возможность настройки вида Gnome/*GTK*-приложений из центра управления KDE. Однако... в Mepis этот пакет установлен по умолчанию, но, судя по всему, не работает. Программы есть, но их невозможно загрузить (об этом говорилось на форумах SimplyMepis).

Не отчаивайтесь: есть другой способ. Установите *gnome-control-center* и используйте *gnome-font-properties*. Запустите программу, набрав в терминале или диалоге запуска (Alt+F2) **gnome-font-properties**, и Вы сможете настроить шрифты Gnome примерно так же, как в KDE.

Этот метод изменит шрифты для текущей сессии, но при перезагрузке изменения будут сброшены. Чтобы их сохранить, выполните следующую команду как обычный пользователь (не root):

```
ln -s /usr/lib/control-center/gnome-settings-daemon ~/.kde/Autostart/
```

Теперь демон настроек Gnome будет загружаться при каждом запуске KDE и применять настройки ко всем *GTK*-приложениям. **МС**

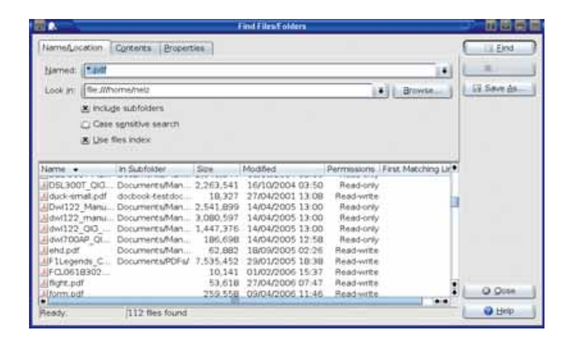
11 Сконфигурируй меня!

В При установке *VNC*-программ на ноутбук с Xubuntu проявилась моя обычная проблема. Я скачал все *VNC*-программы с диска и скопировал их на лэптоп. Затем распаковал архивы *TightVNC* и *VNC* с помощью *tar*, выполнил **cd** в нужную директорию, набрал **./configure**... и увидел сообщение о том, что инструкции не существует. И это обычная картина, когда я ставлю программы с вашего диска, неважно, в Ubuntu или SUSE. Я где-то неправ?

jfl

О По моему скромному мнению, неправы разработчики дистрибутивов. Они считают, что все, что надо, найдется в их репозитарии, а компиляция интересует только разработчиков. Реально же большинству пользователей Linux компилятор бывает нужен – например, для установки свежего драйвера для NVIDIA или сетевой карты. Даже установка двоичного пакета *VMWare* требует сборки соответствующих ядерных модулей. Многие из таких операций также требуют и исходные коды ядра.

Но довольно слов. В Ubuntu Вам нужно установить пакет *build-essential*, включающий все необходимое для сборки программ из исходных текстов. В SUSE понадобится пакет *gcc*. Если Вы захотите установить программу из «исходников» (рано или поздно так и будет), эти пакеты обязательны. Но в данном случае обойдемся без них. В обоих дистрибутивах есть последние версии *TightVNC* в репозитариях или на дисках, а Ubuntu включает и обычный *VNC*. Пока Вам не понадобится самая свежая версия, лучше обойтись пакетами, собранными для вашего дистрибутива, поскольку они протестированы, а об обновлениях Вам сообщат. **НБ**



› ***KFind* находит файлы – да-с! – но не так гибок, как *find*, и не так быстр, как locate.**

12 Верните модем

В Наконец-то я заполучил долгожданную SLED 10 от LXF85. Чудесно! Однако у меня уже есть SUSE 9.3/10.1, и в обеих этих системах мой модем работает нормально, а в SLED 10 – отказывается. Он упорно требует карту Ethernet, которой у меня нет, и говорит, что она не подсоединена. Я сам знаю, что не подсоединена, раз ее нет!

Как заставить модем работать? У меня обычное телефонное соединение, а не широкополосный доступ.

Эрик Джордан [Eric Jordan]

О Устанавливая SLED, Вы видели экран со списком Вашего сетевого оборудования (Ethernet/DSL/модем) и опциями его настройки. Если Ваш ПК предусматривает Ethernet-соединение – а в наше время оно встроено почти во все материнские платы – по умолчанию будет использоваться именно оно. Помните, что SLED – корпоративный дистрибутив, а для предприятий Ethernet – стандарт. Я »

» догадываюсь, что произошло. Настройка по умолчанию для сетевой карты – DHCP. Поскольку карта не соединена с сетью, DHCP терпит неудачу и сообщает об ошибке.

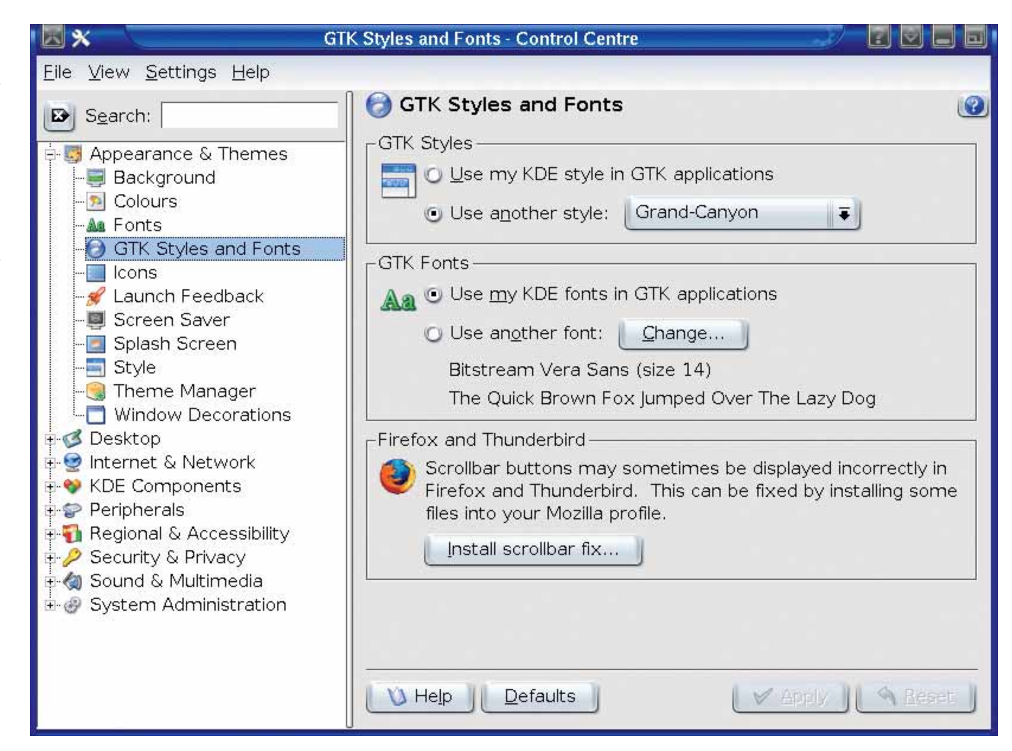
Вам нужно деактивировать Ethernet-соединение, а модем – активировать. Эти действия выполняются в Центре управления. Выберите Сетевые карты (Network Cards), выделите Ваш Ethernet-адаптер и нажмите Удалить (Delete). Карта останется, но будет помечена как не настроенная (Not Configured). Теперь вернитесь в Центр управления и выберите модем. Затем укажите провайдера и введите параметры подключения. НБ

13 Flash For Fedora

В У меня проблема: мой *Firefox* (версия 1.5.0.1) отказывается отображать на web-страницах Flash-ролики. Я использую Fedora Core 5, хотя проблема проявляется практически в любом дистрибутиве Linux. Подскажите, пожалуйста, где взять подключаемый модуль и как «прикрутить» его к браузеру?

Михаил

О Действительно, многие свободные дистрибутивы не включают проигрыватель Adobe Flash по лицензионным соображениям. Альтернатива – использовать платные коробочные версии, хотя, конечно, можно обойтись и без этого. Компания Adobe официально предоставила права на распространение подключаемых модулей Flash в форматах RPM, DEB и ebuild (Gentoo) проекту, сайт которого доступен по адресу: http://macromedia.mplug.org/. Там же можно найти параметры репозиториев YUM и APT/RPM для Fedora Core, подключив которые, Вы сможете установить, а впоследствии – обновлять Flash Player одной командой. В настоящий момент подобным образом распространяется только *Flash Player 7* – находящийся в стадии тестирования *Flash Player 9* будет «упакован» для Linux, как только увидит свет финальная версия. ВС



› **Модуль gtk-qt позволяет настраивать оформление GTK-приложений из центра управления KDE.**

Вопрос-победитель

Иван Алексеевич получает подарочный сертификат на 1000 рублей от интернет-магазина LinuxCenter.Ru! Просим победителя выйти на связь с редакцией: info@linuxformat.ru

Интернет вручную

В С интересом разбираюсь с SUSE Linux 10.1. Поначалу у меня были проблемы с подключением и настройкой ADSL-модема. Сейчас все работает, но подключение и инициализация модема и сетевой платы происходит только при загрузке системы. Я пытался настроить в опциях модема и сетевой платы режим работы «вручную», но ничего не выходит. Подозреваю, что я упускаю какую-то мелочь. Не подскажете, в чем дело?

Бурундаев Иван Алексеевич

О Скорее всего, в Вашей системе отсутствуют инструменты для ручного подключения или у пользователя, от имени которого Вы работаете в SUSE, недостаточно прав работы с ними. Запустите YaST, перейдите в категорию «Сетевые устройства», выберите нужный вариант, дождитесь, пока мастер считает конфигурацию и нажмите кнопку «Редактировать», после чего перейдите на вкладку «Общий». Чуть ниже выпадающего меню с режимом активации (здесь, естественно, следует выбрать пункт «Вручную») находится флажок «Управление пользователем» – установите его.

Собственно подключение и отключение от сети осуществляется посредством утилиты *KInternet*. Не берусь утверждать наверняка, но по-моему, в последних версиях SUSE она больше не устанавливается по умолчанию, поэтому в случае необходимости добавьте эту программу через YaST (она с гарантией присутствует на дистрибутивном диске SUSE). Запустите ее командой *kinternet*, после чего используйте пиктограмму в трее для подключения к Сети. Альтернативный вариант – использовать консольную утилиту *ifup* – значительно менее удобен, зато точно не потребует доустановки дополнительного ПО. ВС

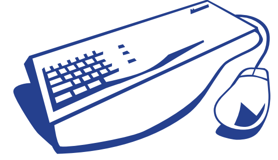
Нужна помощь!

› Для наилучшего ответа на ваш вопрос нам нужно знать как можно больше подробностей. Детально опишите конфигурацию системы. Если вы получили сообщение об ошибке, приведите текст сообщения и точно опишите вызвавшие его действия. Если у вас проблемы с оборудованием, то опишите его. Если Linux уже запущен, то выполните в root-терминале следующие команды и прикрепите к письму файл **system.txt**:

```
uname -a >system.txt
lspci >>system.txt
lspci -vv >>system.txt
```

› Пожалуйста, помните, что сотрудники журнала НЕ являются авторами или разработчиками Linux, любых пакетов или дистрибутивов. Зачастую люди, отвечающие за приложения, выкладывают большую часть информации на web-сайты. Попробуйте почитать документацию!

Мы стараемся ответить на все вопросы. Если вы не нашли ответ на свой, это, возможно, потому, что мы уже ответили на похожий вопрос.



Лучшие новинки открытого ПО на планете

LXF HotPicks



Ричард Драммонд
Ричард – свободный разработчик, писатель и отец двух детей. Он живет в Индиане, США, где отчаянно скучает по British TV, теплому пиву и сосискам.

В ЭТОТ РАЗ ТОЛЬКО ДЛЯ ВАС: GtkLife » Liferea » KBackup » Bluefish » Lost Labyrinth » X-Moto » Xming » HomeBank » MOC » Kdesvn

Клеточный автомат

GtkLife



Версия 5.1 **Сайт** http://ironphoenix.org/tril/gtklife

Жизнь подобна грейпфруту. Может, и так. Но здесь мы говорим не о загадке существования, а об *игре Жизнь* [Life], созданной математиком Джоном Конуэем [John Conway] в 60-х и ставшей популярной благодаря знаменитой колонке Мартина Гарднера [Martin Gardner] в *Scientific American*. Только это не совсем игра, поскольку игроков в ней нет – одни правила. Вы создаете первоначальную конфигурацию на доске и наблюдаете, как «игра» развивается в соответствии с ними.

Жизнь по Конуэю – это клеточный автомат, показывающий, как сложные явления могут проявиться в весьма простой вселенной. Вселенная является двумерной сеткой ячеек, которые могут умирать или рождаться. Состояние каждой ячейки в следующем поколении зависит от ее 8 ячеек-соседей в этом поколении: менее 2 живых соседей – и ячейка умирает от одиночества; более 3 – и она умирает от перенаселения. Новая живая ячейка рождается в любой мертвой ячейке, у которой ровно 3 живых соседа.

Вы можете изучать *Жизнь* при помощи бумаги и ручки или доски для игры Го, но использовать компьютер намного легче: для этого есть инструмент *GTKLife* с открытым кодом. В *GTKLife* вы можете «закрашивать» ячейки мышью, как в монохромном графическом пиксельном редакторе. Можно легко вырезать и вставлять комбинации ячеек или импортировать и экспортировать их в различные файловые форматы. Вы можете перейти к следующим поколениям по шагам или просто нажать Go и смотреть развитие *Жизни* с желаемой скоростью – вплоть до 1000 поколений в секунду на быстрых машинах.

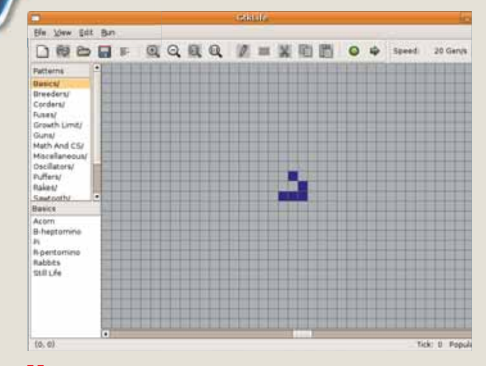
Вы спросите: в чем смысл *Жизни*? Что ж, в ней нет скрытого смысла. Вы должны найти его сами. Ранние исследователи вселенной *Жизни* наслаждались, каталогизируя интересные узоры, открытые ими, и давая им имена вроде «улья», «мигалки» или «жабы». Особо интересны «космические корабли»: рисунки, которые со временем возвращаются к своему исходному состоянию, но в другом месте. Таким образом кажется, что они движутся. Простейшим из них является планер (*см. ниже Шаг за шагом*). Используя некоторые из наиболее интересных свойств планеров, можно строить в *Жизни* логические вентили, и – при достаточном терпении – даже полнофункциональные вычислительные устройства!

«Здесь нет скрытого смысла – вы должны найти его сами.»

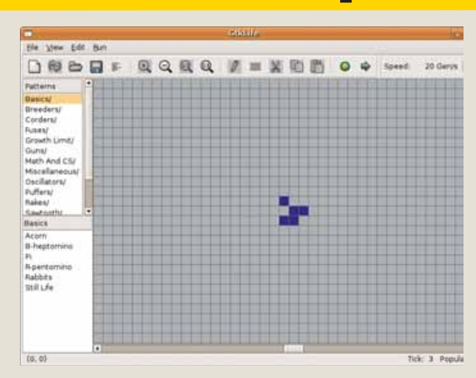
› ***Жизнь*, но не та, что мы знаем. Группа планеров мчится к бесконечности.**

GTKLife поставляется с библиотекой шаблонов для ваших экспериментов, а web-сайт содержит ссылки на дополнительные материалы о *Жизни*.

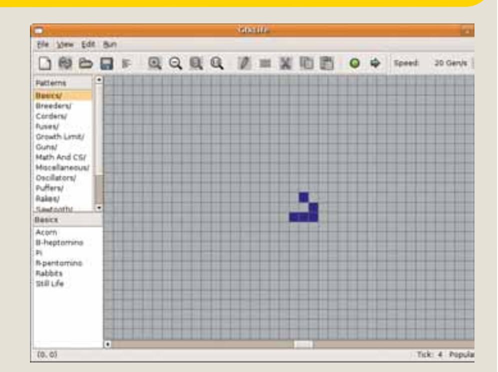
Шаг за шагом: полет планера



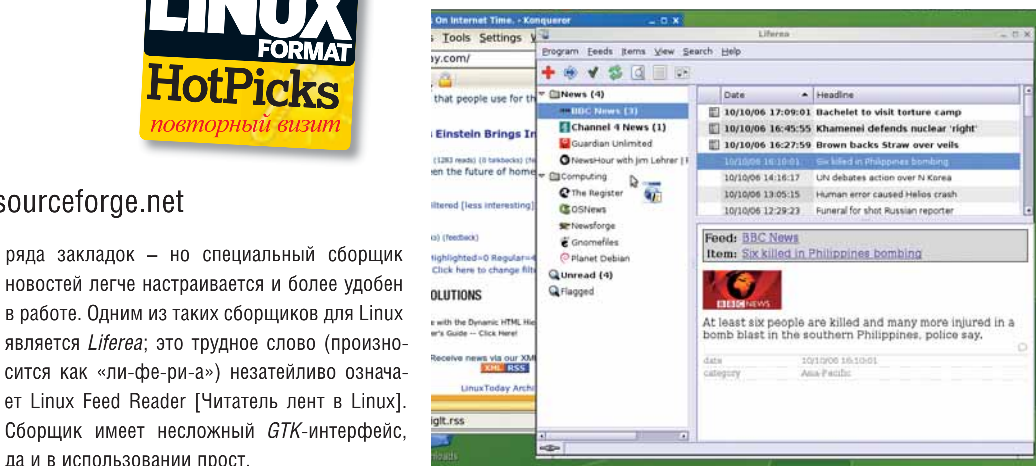
» Постройте ячейки
Выберите New из файлового меню *GtkLife*, увеличьте масштаб и нарисуйте глайдер, как показано на рисунке.



» Запуск
Нажмите иконку Step [Шаг] на панели инструментов для сдвига поколения и отправки планера в полет. Теперь щелкните на Step еще раза два.



» Он летит!
Щелкните на Step еще раз. Исходный рисунок появится вновь, но смещенный на одну ячейку вправо и вниз. Нажмите Go [Запуск] и созерцайте, как он улетает в бесконечность.



Сборщик новостей

Liferea

Версия 1.0.23 **Сайт** http://liferea.sourceforge.net



Мы живем в мире «по требованию». Ну или нам так часто говорят. Новости нам подавай немедленно; мы не хотим сами гоняться за ними. Для тех, кто алчен до событий, такую связь по запросу обеспечивает инструмент-сборщик новостей, он же читатель новостных лент. Если вы все еще каждое утро завтракаете с газетой, знайте: сборщики новостей – это программы, позволяющие пользователям подписываться на новостные ленты web-сайтов и автоматически загружать свежие заголовки в удобной для чтения форме. Технологическая основа этого чуда – файловый формат RSS на базе XML; RSS можно расшифровать как Rich Site Summary [Подробный обзор сайта], или Really Simple Syndication [Действительно простой сбор данных], или любыми другими подходящими словами – придумайте сами. На каком бы web-сайте вы ни увидели символ RSS, он означает новостную ленту, к которой можно подключиться.

Браузеры часто имеют поддержку RSS – Firefox, например, представляет ленту в виде ряда закладок – но специальный сборщик новостей легче настраивается и более удобен в работе. Одним из таких сборщиков для Linux является *Liferea*; это трудное слово (произносится как «ли-фе-ри-а») незатейливо означает Linux Feed Reader [Читатель лент в Linux]. Сборщик имеет несложный *GTK*-интерфейс, да и в использовании прост.

По умолчанию *Liferea* состоит из трех панелей: слева список подключенных лент, размещенных в указанных пользователем папках; вверху заголовки статей текущей выбранной ленты; внизу – текущая статья. Последняя может отображаться по вашему выбору через виджет *GTKHTML* или встроенный HTML-движок *Mozilla*. Виртуальные каталоги позволяют группировать статьи по определенному пользователем фильтру: например, по статусу «выделено» или «не прочитано», или по заголовку статьи. Ссылки в статьях можно просматривать во внешнем браузере – *Liferea* имеет встроенную поддержку основных браузеров Linux – а можно настроить *Liferea* на открытие и отображение ссылок в окне *Liferea*.

› Просто перенесите иконку RSS с web-сайта из вашего браузера и бросьте ее в окно *Liferea*, чтобы подписаться на ленту новостей.

> **«Специализированный сборщик новостей более удобен в работе.»**

Последний раз мы рассматривали *Liferea* два года назад, в LXF60. С того времени проект выпустил стабильный релиз и проделал более 20 пересмотров этой стабильной ветки. Текущая версия – зрелый и устойчивый продукт. А если вы рветесь на передний край, то для тестирования доступна также нестабильная, разрабатываемая версия.

Инструмент резервирования

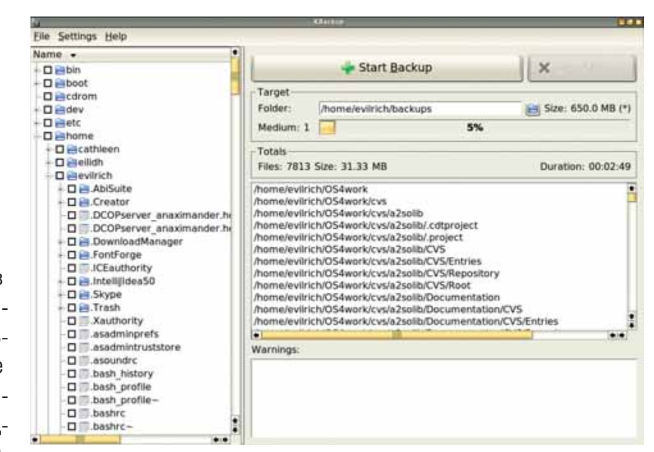
KBackup

Версия 0.5.1 **Сайт** www.kde-apps.org/content/show.php?content=44998

Данные надо резервировать. Истина избитая, мы знаем, но, по нашему же опыту, однажды вы можете сильно пожалеть, что не делали этого. Резервное копирование подобно страхованию. Никто не ожидает и тем более не жаждет, чтоб оно пригодилось, но случись что – и вы будете ох как рады припасенному бэкапу. Для новичков в Linux трудность в том, что понятных и легких в использовании инструментов мало, и они очень различаются. Сложных приложений резервного копирования хватает, но похоже, никто не позаботился о пользователе ПК, и это удивительно в свете всех последних усилий продвижения Linux в качестве настольной платформы. *KBackup* пытается изменить ситуацию.

Как подобает инструменту для неопытного пользователя, *KBackup* придерживается простоты. Он предоставляет графический интерфейс, позволяющий пользователю выбирать файлы и каталоги для резервирования, и использует вездесущий tar-файл в качестве хранилища с сжатием gzip или bzip2. Вы можете хранить свои резервные копии в локальном каталоге (используя, скажем, примонтированное устройство), на сетевом SMB-диске или на удаленном ресурсе. *KBackup* не утруждается модным инкрементным резервированием или автоматизацией и даже не поддерживает восстановление с резервной копии. Это преимущество использования формата tar: для распаковки данных из tar-архивов инструментов и так полно.

KBackup, по сути, лишь немногим больше, чем этот знаменитый архиватор, но для резервирования он удобнее, благодаря кое-каким трюкам. Во-первых, он автоматически нумерует архивы при помощи определенного пользователем префикса и даты, и вам не нужно делать это самостоятельно. Во-вторых, он поддерживает автоматическое разбиение резервной копии на части определенных размеров: удобная функция при необходимости сохранения архива на сменных носителях вроде CD-R. Размер частей настраивается пользователем; кстати, полезно было бы добавить некоторые распространенные размеры носителей по умолчанию. В-третьих, *KBackup* может сохранять сессии резервирования в виде профилей, и потом их можно вновь загрузить для следующего резервирования тех же файлов и каталогов. Это здорово экономит время при регулярном резервировании.



› Это может делать каждый: регулярное создание резервных копий.

Да, *KBackup* чрезвычайно примитивен. Хорошо бы увидеть побольше добавочных функций без снижения простоты. Но, в том виде как есть, это определенно шаг в верном направлении к созданию инструмента, скроенного под пользователей настольных машин.

Web-редактор

Bluefish



Версия 1.0.6 **Сайт** http://bluefish.openoffice.nl

Для web-разработки можно использовать обычный текстовый редактор, а можно выбрать визуальную среду разработки на все руки. Но имеется и третий вариант, который лежит где-то посередине: web-редактор – текстовый редактор, приспособленный для web-программирования. Это компромисс, сочетающий гибкость текстового редактора с возможностями быстрой разработки, присущими IDE. *Bluefish* – зверь из этой породы. Это не инструмент для новичков, но в руках умелого web-разработчика он способствует ошеломляющему ускорению, позволяя при этом полностью управлять разработкой web-страницы.

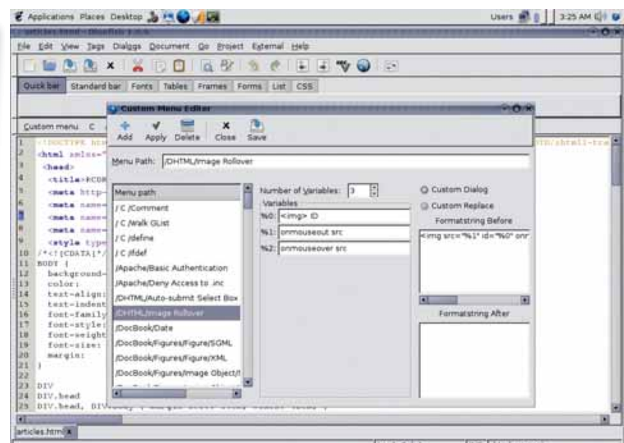
Bluefish, основанный на *GTK*, преподносится как редактор «Видите то, что вам и нужно» (WYSIWYN). Он не претендует на роль визуальной среды. В отличие от своего KDE-соперника, *Quanta*, он даже не предоставляет функции предпросмотра, показывающей, как страница будет выглядеть при отображении в браузере: эта задача переложена на внешние приложения. А вот что он предоставляет, так это ввод исходного кода для страниц, быстрого и точного.

В основе *Bluefish* – старый добрый текстовый редактор: он поддерживает подсветку синтаксиса, неограниченные откат и повторение, множество кодировок, сглаживание шрифтов и возможность редактирования множества файлов в отдельных вкладках редактора (разработчики заявляют, что 500 документов обрабатываются запросто). Но далее начинаются различия. *Bluefish* может открывать файлы при помощи любого метода, поддерживаемого gnome-vfs – в том числе, удаленно. В зависимости от ваших настроек, сюда могут входить файлы, доступные по HTTP, FTP, WebDAV и Samba. По умолчанию поддерживается подсветка синтаксиса для большинства известных языков web-разработки, включая HTML, XML, CSS, JavaScript, PHP, JSP, ColdFusion, Perl, Python и Java. Он даже поддерживает Pascal: вдруг какой-нибудь извращенный тип примется разрабатывать на этом языке CGI-инструменты!

Движок подсветки синтаксиса настраивается на любой другой язык через Perl-совместимые регулярные выражения. Редактор *Bluefish* также включает проверку орфографии HTML-страниц и предоставляет удобные функции вроде автоматического завершения HTML-тэгов.

RAD-инструменты

Не исключено, что вы покамест не в восторге. Этот *Bluefish* не более чем заурядный редактор, скажете вы. Не спешите с выводами! На страницу можно вставлять все часто используемые простые HTML-тэги – или путем выбора элемента меню, или при помощи клавиатурных комбинаций. Тэги посложнее вставляются при помощи диалогов, и вновь большинство из них связаны с горячими клавишами. Имеются диалоги для генерации заголовков страницы (включая тэги META), фреймов, форм, таблиц и CSS-стилей. Более изощренные диалоги позволяют вставлять изображения, при желании – с автоматической генерацией миниатюр.



› *Bluefish* можно полностью перестроить «под себя».

Имеются и другие клевые вещи: кроме встроенных диалогов, поддерживаются диалоги, определяемые пользователем. Вы можете добавить свой собственный инструмент для генерации шаблона кода, способный запрашивать параметры у пользователя. Добавляемые вами шаблоны доступны на панели пользовательского меню *Bluefish*, и имеется редактор для их разработки, так что никакой правки файлов конфигурации не потребуется. Стандартная установка включает инструменты для генерации JavaScript-сценариев, стандартных кусков кода на PHP и даже операторов SQL.

Высшие вызовы

Как мы говорили, *Bluefish* не имеет встроенной функции предпросмотра HTML, а также не имеет проверки синтаксиса HTML и XML или инструмента форматирования кода. Зато в нем есть прекрасный механизм вызова внешних инструментов. Вы настраиваете их сами, но стандартные настройки уже поддерживают использование weblint для проверки HTML или tidy для форматирования HTML-кода. Вывод таких внешних инструментов перехватывается и отображается в окне *Bluefish*.

А в боковой панели *Bluefish* вы найдете файловый менеджер, просмотрщик документации и редактор закладок. Последний предоставляет быстрые ссылки на HTML-тэги, CSS-атрибуты и функции PHP и Python. Включены средства поиска, а также механизм вставки указанных тэгов или функций в ваш код. Закладки позволяют быстро возвращаться к определенной пользователем точке в вашем файле. В руках опытного пользователя *Bluefish* превращается в мощную среду редактирования.

Исследуем интерфейс Bluefish

Окно редактирования
Bluefish поддерживает множественные вкладки, сглаживание шрифтов, подсветку синтаксиса и проверку на ошибки HTML файлов.

Меню приложения
Здесь можно обратиться к большинству функций *Bluefish*. Сюда включены большинство HTML-тэгов и объектов, а также возможность вызова внешних инструментов.

Панель HTML
На этой панели доступно большинство HTML-тэгов и диалогов. Вы можете настроить панель быстрого доступа, просто добавив любую кнопку.

Пользовательское меню
Предлагает диалоги для создания шаблона кода.

Боковая панель
Файловый менеджер *Bluefish*, браузер документации и менеджер ссылок – все живут здесь.

Окно вывода
Вывод внешних команд, например, результат использования weblint для проверки вашего HTML или использование php для обработки файла, показывается здесь.

HotGames Игры и развлекательные приложения

RPG

Lost Labyrinth

Версия 2.8.0 **Сайт** www.lostlabyrinth.com

Компьютерные игры в наши дни требуют от разработчика большой отдачи. Удивляться тут нечему. Когда отстегиваешь 30 или 40 фунтов, хочется получить за свои денежки нечто солидное. А если вам нужна просто блиц-забава на свободные 15 минут, а не долгосрочная эпопея, играть в которую и за неделю не выучишься? Тут-то и приходят на помощь программисты Open Source.

Lost Labyrinth обозначена как «ползалка по подземельям на время перекура». Это игра типа Rogue, задуманная как быстрый, забавный и (главное) короткий процесс. К примеру, ваш герой владеет лишь навыками, выбранными при старте; он или она не могут изучить новые. К тому же, опыт добывается путем исследования, а не путем сплошного монстроубийства. С хитрой и яркой графикой и легкой в использовании системой управления при помощи мыши или клавиатуры, *Lost Labyrinth* готов к игре немедленно. И не принимайте простоту за скудость средств. Кроме обычной толпы монстров для битья и объектов для сбора, в изобилии имеются «сюрпризы», появляющиеся случайным образом и освежающие игру. Среди них библиотека, богатая древними свитками; таверна, где можно нанять компаньона себе в помощь; и ведьмин сад с волшебными грибами и лекарственными травами (надо только избавиться от самой ведьмы).

Игра *Lost Labyrinth* написана на несвободном PureBasic, поэтому большинство пользователей не захотят компилировать ее самостоятельно. К счастью, авторы изготовили пакеты RPM и Debian, доступные для загрузки. Мы протестировали Deb и обнаружили, что если вы хотите иметь возможность сохранять игры и лучшие результаты, то вам необходимо выполнить следующее (от имени root):

```
chgrp -R games /usr/games/laby
chmod -R g+w /usr/games/laby
```

Не считая этого недостатка, *Lost Labyrinth* – качественный продукт, напрашивающийся на более, чем один запуск.

> «Опыт добывается путем исследований, а не путем монстроубийства.»



› Перебежать по горячей лаве, чтоб урвать еще кусок Жезла Последнего Завета… а время есть только на бисквит.

Мотокросс

X-Moto

Версия 0.2.2 **Сайт** http://xmoto.sourceforge.net

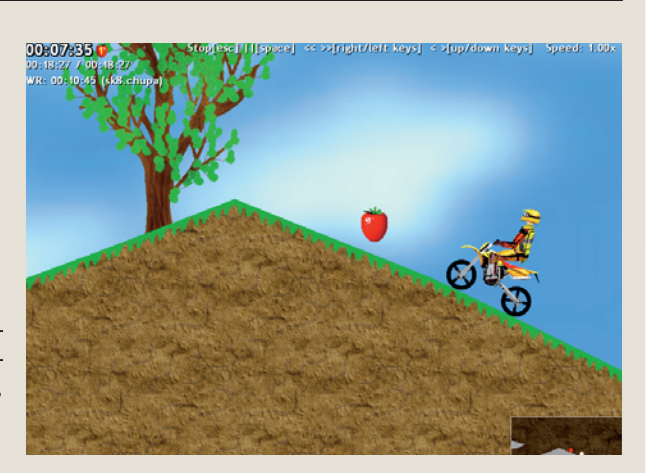
В восьмидесятых годах существовало телевизионное мотокросс-шоу под названием *Kick Start*, в котором мотоциклисты преодолевали полосу с серьезными препятствиями, на время. Это действо было отражено в неофициальной компьютерной игре легендарного Шона Саузерна [Shaun Southern] для восьмибиток, под названием *Kikstart*. Если вам нравилось одно из этих двух, вы полюбите *X-Moto*. Здесь нет ни обязательной в *Kikstart* игры двух игроков, ни жуков-Фольксвагенов для прыжков, но вам доставит много удовольствия выполнение невероятных трюков на мотоцикле.

Цель *X-Moto* – как можно быстрее завершить уровень. Уровни – это не рукотворная полоса препятствий, как в *Kikstart*: ваша основная задача – справиться с трассой. Вы должны провести мотоцикл по склонам, ямами и безднам, сквозь пещеры. Чтобы сделать это с минимальным числом сломанных конечностей, вы должны прежде всего мастерски управлять мотоциклом. Физика играет в *X-Moto* наиважнейшую роль. Гравитация, инерция и трение поверхности, по которой вы движетесь – все влияет на поведение вашего мотоцикла, а у вас только базовое управление: ускорение, торможение и поворот влево и вправо. Сперва вы намучаетесь, пытаясь хотя бы удержать колеса на грунте. К счастью, прилагается несколько обучающих уровней. Есть управление джойстиком (в порядке эксперимента), но вы, вероятно, предпочтете управлять клавиатурой, она буквально сверхчувствительна.

У *X-Moto* процветающее сетевое сообщество, там вы можете найти дополнительные уровни, статистику лучших результатов, wiki и форумы. Если вы захотите добавить свой собственный сложный ландшафт, прилагается редактор уровней.

Сборка *X-Moto* стандартна, если у вас установлены все зависимости. Потребуются *SDL*, *OpenGL*, *Lua*, ODE (Open Dynamics Engine). Детали см. в документации. Многие дистрибутивы, однако, создают свои пакеты *X-Moto*. И все же мы хотим надеяться, что скоро появится *Junior X-Moto*...

> «Сперва вы намучаетесь, пытаясь хотя бы удержать колеса на грунте.»



› Функция повтора в *X-Moto* позволяет упиваться победой вновь и вновь и…

X-сервер

Xming

Версия 6.9.0.18 **Сайт** www.straightrunning.com/XmingNotes

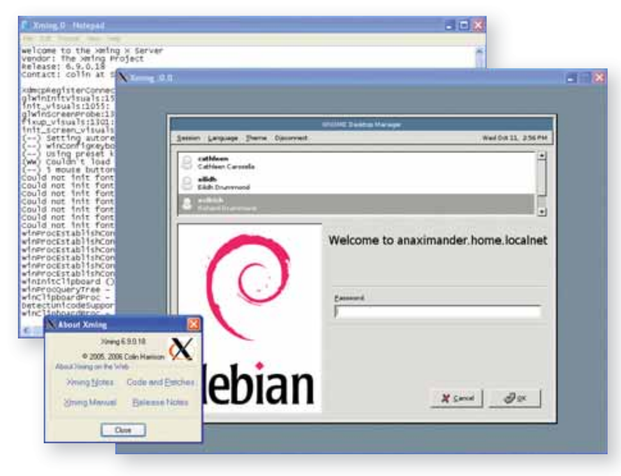
Xming не имеет ничего общего с неким фантастическим сериалом или китайскими династиями: на самом деле это X-сервер для Windows, порт сервера *X.org*. Минуточку, кричите вы, вроде уже есть порт X-ов в проекте *Cygwin*? Ну да, есть. Но *Xming* создан с использованием *MinGW*-версии GCC (родного Win32-порта GCC) вместо использования уровня POSIX-совместимости, *Cygwin*. Это делает установку проще и быстрее, да *Xming* и полегче.

Программа распространяется в виде самораспаковывающегося исполняемого файла. Просто загрузите и запустите его; заработает установщик, а вы следуйте подсказкам. Легче легкого. Вот настолько *Xming* прост. Настройки он не требует, хотя имеется несколько опций, с которыми можно побаловаться. Как и сервер *Cygwin*, *Xming* может работать в режиме 'rootless'. В обычном режиме его экран отображается в окне; root-окно – обычно ваш рабочий стол – занимает все окно *Xming*, и окна, принадлежащие X-клиентам, живут только внутри окна *Xming*. В rootless или многооконном режиме root-окна нет, и окна X-клиентов создаются так, как будто они – родные приложения. Этот режим позволяет X-клиентам выглядеть привычно на рабочем столе Windows.

Xming включает графический инструмент под названием *XLaunch* для настройки и использования профиля *Xming*. Он позволяет выбирать между многооконным (rootless) и однооконным (обычным) режимами и имеет несколько других опций: например, запускать ли локальную X-сессию или подсоединиться к удаленной машине, используя XDMCP. *XLaunch* пока не поддерживает всех опций *Xming*. Например, вы не можете выбрать размер экрана в однооконном режиме. Но можно указать произвольные опции командной строки для *Xming*, так что просто прочтите man-страницы *Xming* и укажите необходимые параметры. Чтобы установить, например, размер экрана 800x600, добавьте

```
-screen 0 800 600
```

> «Выберите локальную X-сессию или удаленную машину через XDMCP.»



› Удаленный доступ к Linux-компьютеру – лишь одно из многих применений *Xming*.

Поскольку *Xming* – порт современного X-сервера, то не удивительно, что он включает современные расширения X, например, *Render* и *XrandR*. Правда, расширение *XRandR* на данном этапе большей частью бесполезно, поскольку *Xming* сейчас не поддерживает изменения размеров экрана после запуска сервера.

Персональный финансовый менеджер

HomeBank

Версия 3.2 **Сайт** http://homebank.free.fr

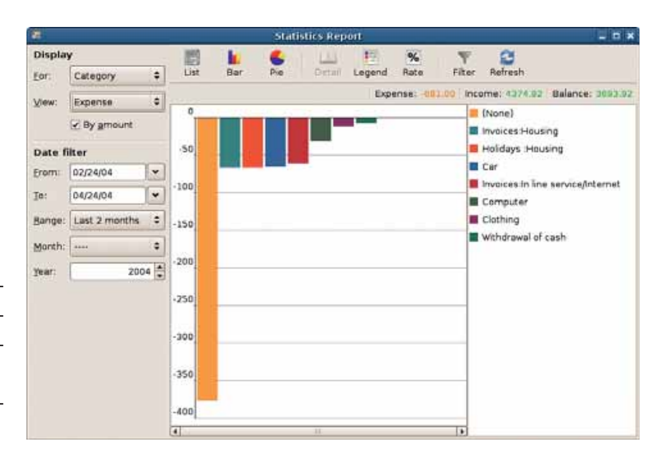
Владельцы первых персональных компьютеров изо всех сил пытались оправдать свою покупку. Самое частое оправдание – извините, применение – для этой дорогой груды кремния, меди и пластика было ведение баланса чековой книжки. Реальность, как обнаружили пионеры цифровой эпохи, была в том, что карандаш и бумага были быстрее и сговорчивее. Зато сегодня, когда компьютеры намного мощнее и легче в использовании, такие задачи вполне возможны, как доказывает приложение *HomeBank*.

HomeBank – *GTK*-приложение для управления вашим персональным банковским счетом, сравнительно новое для платформы Linux, но на самом деле являющееся авторским портом Максима Дуайена [Maxime Doyen] давнишней программы для Amiga. *HomeBank* существует уже более десяти лет, поэтому если вы заинтересованы в долговечности – а в данной предметной области это немаловажно – то *HomeBank* выглядит хорошим выбором.

Интерфейс *HomeBank* подобен многим бухгалтерским программам. Хотя имеется несколько отличий от обычной бухгалтерской терминологии, к которым придется привыкнуть – например, *HomeBank* называет транзакции по счету «операциями», а повторяющиеся транзакции (например, прямой дебит для вашей ипотеки или ренты), почему-то, «архивами»; но документация поможет преодолеть эти начальные барьеры.

Сила приложений, подобных *HomeBank*, в том, что становится легко отследить, куда уходят ваши деньги. Вы создаете пользовательские категории – например, платежи по чекам, рента, коммунальные платежи, еда, одежда и так далее – а затем каждую транзакцию, добавляемую по счету, относите к одной из них. Онлайн-средства многих банков предоставляют сходные возможности, но одиночные приложения более искусны, а

> «Если вас интересует долговечность – вот хороший выбор.»



› Визуальные отчеты и фильтры *HomeBank* покажут, куда утекли ваши наличные.

HomeBank выделяется даже и среди них, особенно инструментом создания отчетов. Хотите знать, сколько вы ежемесячно проедали в прошлом году? Как это соотносится с аппетитом предыдущего года? *HomeBank* легко ответит на подобные вопросы и представит отчеты в текстовом или графическом виде. Другая полезной функция – составление бюджета, позволяющее установить лимиты расходов по каждой категории.

В настоящий момент нет двоичных файлов *HomeBank*, но программу легко собрать вручную, поскольку она имеет всего одну зависимость – *GTK*. Сейчас Дуайен ищет разработчиков: нужна помощь в адаптации пакетов *HomeBank* к конкретным дистрибутивам.

Музыкальный проигрыватель

MOC

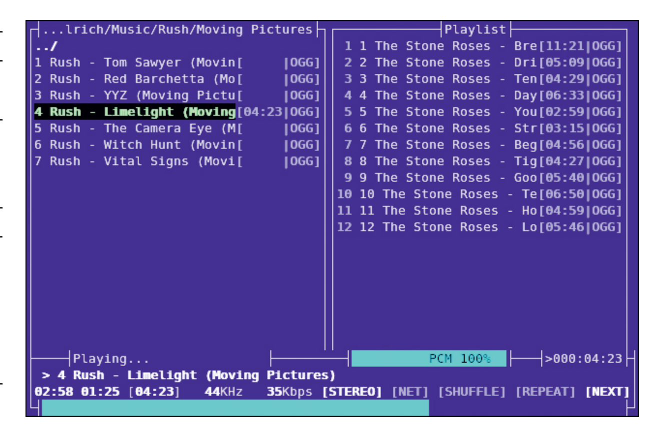
Версия 2.4.1 **Сайт** http://moc.daper.net

Вычурный графический интерфейс не делает приложение удобным. Возьмите, к примеру, *MOC* (Music On Console – музыка в консоли). Это полноценный музыкальный проигрыватель, который делает то же, что и его тяжеловесные собратья, но только через интерфейс пользователя *ncurses*. Да, он не визуализирует проигрываемый звук – ну и кому это надо?

MOC умеет озвучивать несколько аудиоформатов, включая MP3, Ogg и Flac. Он поддерживает плей-листы и потоковое вещание в web, и имеет встроенный регулятор громкости; он даже буферизует следующий трек в памяти, предотвращая обрывы звука. Более того, если консоль понадобится вам для других задач, благодаря продуманному дизайну интерфейса пользователя вы можете его отключить, оставив проигрывание звука в фоне, а для возобновления подсоединения просто перезапустить программу.

Интерфейс *MOC* быстр и легок в использовании, поскольку создавался с оглядкой на популярный файловый менеджер Midnight Commander. Встроена подсказка, описывающая управление с клавиатуры. Сборка *MOC* требует только простых **configure** и **make**. Большинство дистрибутивов уже содержат *ncurses*, но убедитесь, что у вас есть библиотеки от проектов Ogg Vorbis, *Flac* и *FFmpeg* для полной поддержки всех форматов.

› Плей-лист, прогресс-индикатор и темы – MOC делает все, что полагается графическому аудиопроигрывателю.



Клиент Subversion

Kdesvn

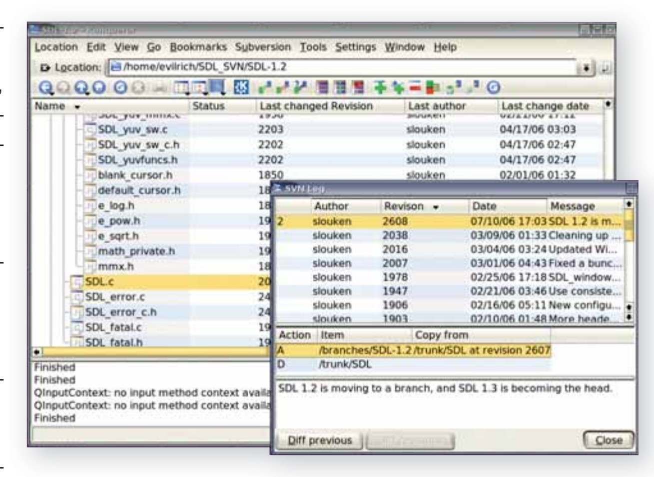
Версия 0.10.0 **Сайт** www.alwins-world.de/programs/kdesvn

Система управления версиями CVS хорошо послужила сообществу Open Source, но за ее место теперь борются различные заменители, исправляющие многие ошибки. Если смотреть на них с позиций популярности, то фаворитом окажется *Subversion*, и многие проекты высокого полета уже перешли на этого убийцу CVS. Среди многих желанных функций *Subversion* имеется API для разработчиков собственных клиентов – они не должны возиться с обеспечением взаимодействия с клиентом командной строки. *Kdesvn* и есть такой клиент *Subversion*, встроенный в платформу KDE. «Это действительно клиент, а не оболочка инструмента командной строки», поясняет разработчик Райко Албрехт [Rajko Albrecht].

Kdesvn выполняет все, что требуется от клиента *Subversion*, плюс предоставляет высокую степень интеграции с рабочим столом KDE. Можно выписывать (check-out) код из репозиториев, фиксировать (check-in) изменения, просматривать журналы изменений, сравнивать версии, и многое другое. Его основная функциональность реализована в виде kio_slave и KPart, которые могут повторно использоваться в других проектах KDE, и он может использовать *Kompare* для графического отображения различий.

Сборка *Kdesvn* может слегка озадачить. Как вы, вероятно, знаете, KDE переходит с autotools на *cmake*. Что ж, *Kdesvn* уже на стороне победителя. Если вы не знакомы с *cmake*, лучше почитать о нем перед компиляцией *Kdesvn*. LXF

› *Kdesvn* может работать сам или как расширение к *Konqueror*.

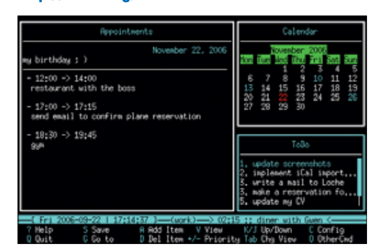


Также вышли

Новые и обновленные приложения, также заслуживающие внимания

Aria2 0.8.1 Программа для закачки, поддерживающая HTTP, FTP и BitTorrent http://aria2.sourceforge.net

Calcurse 1.6 Персональный планировщик на основе Curses http://culot.org/calcurse



› Консольная Calcurse.

Comix 3.6 Просмотрщик изображений, особенно подходящий для чтения комиксов http://comix.sourceforge.net

FBReader 0.7.4k Читалка электронных книг для PDA и настольных компьютеров http://only.mawhrin.net/fbreader

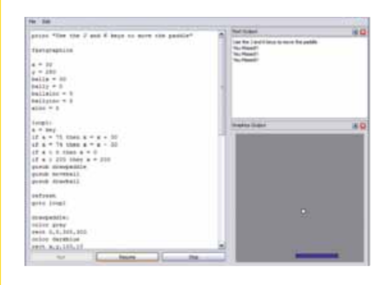
GNU Solfege 3.6.0 Инструмент тренировки слуха со множеством упражнений www.solfege.org

Htop 0.6.4 Просмотрщик процессов – лучше, чем top http://htop.sourceforge.net

Htpasstool 1.0.1 Настройка htaccess через web http://patch.be/htpasstool

Kalva 0.8.81 Легкий в использовании видеомагнитофон KDE. http://developer.berlios.de/projects/kalva

KidBasic 0.3 Простая IDE для детей http://kidbasic.sourceforge.net



› KidBasic: старая школа кодирования.

LXPanel 0.1.1 Легковесная X11 панель рабочего стола http://lxpanel.sourceforge.net

Python Kye 0.9.1 Головоломка по мотивам Sokoban http://games.moria.org.uk/kye/pygtk

UPX 2.90 Архиватор исполняемых файлов http://upx.sourceforge.net

XCB 1.0 rc2 Прямое использование X из C http://xcb.freedesktop.org

XML Copy Editor 1.0.7.7 Быстрый XML-редактор с валидацией http://xml-copy-editor.sourceforge.net

Xpad 2.12 Записки на рабочем столе http://xpad.sourceforge.net

LXF DVD 87/88

Mandriva опять станет нас удивлять, а еще есть журнал в PDF!



Майк Сондерс любовно подбирает содержимое диска *Linux Format*, а также поддерживает сайт **www.linuxformat.co.uk**.

И сладко, и на халяву

За несколько лет жизни в Linux легко забыть, как все сложно на «другой стороне». Если пользователю Джо, например, перепадет «голый» компьютер б/у, без ОС, то ему прежде всего придется раздобывать нехилую сумму денег на Windows, чтобы его машина могла сделать хоть что-нибудь. А когда появившаяся Vista постепенно приберет к рукам рынок OEM, разоряться на *апгрейд* придется всем нынешним пользователям Windows XP.

Если, конечно, пользователь Джо не перейдет на Linux. Все (и совершенно правильно) рассуждают о таком аспекте Linux, как свобода распространения – возможность доступа к исходному коду и его копированию; но не менее важен и тот факт, что он еще и бесплатен «как пиво». Он дает людям всей планеты шанс исследовать весь мир компьютерных технологий, независимо от рынка, и пользоваться мощной ОС.

В этом месяце на нашем DVD представлен превосходный новый релиз Mandriva Free – это дистрибутив, равно подходящий и новичкам в Linux, и ветеранам. Одна из самых потрясающих новых функций – это трехмерный рабочий стол *AIGLX* : и Vista, и OS X тут определенно есть чему позавидовать. Прибавьте к этому большой выбор программ – и вот перед вами дистрибутив, способный украсить любую настольную систему. Вдобавок он распространяется свободно, так что если вы не в настроении использовать Mandriva, вы можете передать его другу или коллеге и поделиться с ними всеми преимуществами использования этой отличной ОС! Да не забудьте просмотреть материалы журнала в формате PDF – подробности на стр.117.

mike.saunders@futurenet.co.uk

Краткое содержание DVD



Информация о диске

Внимательно прочтите это перед тем, как использовать DVD-диск.

ЧТО-ТО ПОТЕРЯЛИ?

Часто случается, что новые программы зависят от других программных продуктов, которые могут не входить в текущую версию вашего дистрибутива Linux.

Мы стараемся предоставить вам как можно больше важных вспомогательных файлов. В большинстве случаев, последние версии библиотек и другие пакеты мы включаем в каталог «Essentials» (Главное) на прилагаемом диске. Поэтому, если в вашей системе возникли проблемы с зависимостями, следует заглянуть именно туда.

ФОРМАТЫ ПАКЕТОВ

Мы стараемся включать как можно больше различных типов установочных пакетов: RPM, Deb или любые другие. Просим вас принять во внимание, что мы ограничены свободным пространством и доступными бинарными выпусками программ. По возможности, мы будем включать исходные тексты для любого пакета, чтобы вы смогли собрать его самостоятельно.

ДОКУМЕНТАЦИЯ

На диске вы сможете найти всю необходимую информацию о том, как устанавливать и использовать некоторые программы. Пожалуйста, не забывайте, что большинство программ поставляются вместе со своей документацией, поэтому дополнительные материалы и файлы находятся в соответствующих директориях.

ЧТО ЭТО ЗА ФАЙЛЫ?

Если вы новичок в Linux, вас может смутить изобилие различных файлов и расширений. Так как мы стараемся собрать как можно больше вариантов пакетов для обеспечения совместимости, в одном каталоге часто находятся два или три файла для различных версий Linux, различных архитектур, исходные тексты и откомпилированные пакеты. Чтобы определить, какой именно файл вам нужен, необходимо обратить внимание на его имя или расширение:

имя_программы-1.0.1.i386.rpm – вероятно, это бинарный пакет RPM, предназначенный для работы на системах x86;

имя_программы-1.0.1.i386.deb – такой же пакет, но уже для Debian;

имя_программы-1.0.1.tar.gz – обычно это исходный код;

имя_программы-1.0.1.tgz – тот же файл, что и выше по списку: «tgz» - это сокращение от «tar.gz»;

имя_программы-1.0.1.tar.bz2 – тот же файл, но сжатый bzip2 вместо обычного gzip;

имя_программы-1.0.1.src.rpm – также исходный код, но поставляемый как RPM-пакет для упрощения процесса установки;

имя_программы-1.0.1.i386.FC4.RPM – бинарный пакет RPM для x86, предназначенный специально для операционной системы Fedora Core 4;

имя_программы-1.0.1.ppc.Suse9.rpm – бинарный пакет RPM, предназначенный специально для операционной системы SUSE 9.x PPC;

имя_программы-devel-1.0.1.i386.rpm – версия для разработчиков.

Если диск не читается...

Это маловероятно, но если все же прилагаемый к журналу диск поврежден, пожалуйста, свяжитесь с нашей службой поддержки по электронной почте: disks@linuxformat.ru

СТОРОНА 2

fedora Core 6

СТОРОНА 1

Mandriva Free 2007

ДВОЙНОЙ ДИСК

ДВОЙНОЙ НОМЕР

А ТАКЖЕ:
- Blender – открытый 3D-редактор
- KDE4 – SVN-срез популярной рабочей среды
- MonoDevelop – среда разработки для Mono
- Liferea – RSS-агрегатор
- Bluefish – редактор HTML/XML
- X-Moto – симулятор мото-триала с необычной физикой
- QuiteInsane – фронтэнд для SANE

DVD 87/88

ЯНВАРЬ 2007

LINUX FORMAT









Содержание Linux Format DVD

Январь 2007

Сторона 1

Рабочий стол
Blender – открытый 3D-редактор
Day_Planner – простой органайзер
GOCR – система распознавания текста
Mirage – просмотрщик изображений на GTK+
Scratchpad – текстовый редактор для GNOME
TkDVD – фронтэнд к программам записи DVD
Xpdf – открытый просмотрщик PDF-файлов
KDE4 – SVN-срез популярной рабочей среды

Разработка
Conglomerate – редактор XML
Harmony – реализация Java 5
Mono – открытая реализация .NET
MonoDevelop – среда разработки для Mono
Qt – фреймворк для разработки кроссплатформенного ПО
TkDiff – графический интерфейс к diff
Valgrind – утилита для слежения за утечками памяти
xmlto – интерфейс к набору XSL

Дистрибутивы
Mandriva 2007 Free – свободная версия популярного французского дистрибутива Linux

Игры
Balder2D – двумерный шутер
Bouncy – почувствуйте себя кроликом! (:
Cultivation – необычная стратегия
Luola – клон V-Wing
Toy_Cars – двумерные гонки с необычной физикой

Справка
RUTE – руководство по администрированию GNU/Linux

Hot_Picks
Bluefish – редактор HTML/XML
HomeBank – программа для управления финансами
Kbackup – простая утилита для создания резервных копий
KDESvn – интерфейс к Subversion для KDE
Liferea – RSS-агрегатор
Lost_Labyrinth – ролевая игра
MOC – консольный аудиоплеер
X-Moto – симулятор мото-триала с необычной физикой
Xming – X-сервер для Windows

Интернет
GtkDC – клиент для сетей DirectConnect
KFTPGrabber – FTP-клиент для KDE
Twinkle – SIP-телефон

Офис
Faces – продвинутый менеджер проектов
Koffice – офисный пакет для KDE
OpenOffice.org – кроссплатформенный офисный пакет

Безопасность
AWStats – утилита для генерации статистики
ClamAV – свободный антивирус
Ethereal – свободный сниффер
GnuPG – альтернатива PGP
Nmap – сканер портов
PortSentry – сервис для защиты от сканирования
Shorewall – утилита для настройки пакетного фильтра
Snort – система обнаружения вторжений
Tripwire – утилита для мониторинга систем.

Звук
Dino – MIDI-сервенсер
KtabEdit – редактор табулатур для KDE
Rhythmbox – продвинутый аудиоплеер с менеджером коллекции
Schism Tracker – редактор трекерной музыки

Система
Mergeant – фронтэнд к Gnome-DB
Moodss – графическая программа для мониторинга системы
QuiteInsane – фронтэнд для SANE
SANE – комплект программ для работы со сканерами

Сторона 2

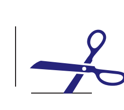
Fedora Core 6

Комментарии? Присылайте ваши мысли и предложения по электронной почте: info@linuxformat.ru
Пожалуйста, ознакомьтесь с опубликованной в журнале инструкцией перед использованием данного диска.

Настоящий диск тщательно тестировался и проверялся на всех стадиях производства, однако, как и в случае с любым новым ПО, мы рекомендуем вам использовать антивирусный сканер. Мы также рекомендуем всегда иметь под рукой актуальную резервную копию данных вашего жесткого диска. К сожалению, редакция Linux Format не может принимать на себя ответственность за любые повреждения, разрушения или иные убытки, которые может повлечь за собой использование этого DVD, представленных на нем программ или данных. Перед тем, как устанавливать какое-либо ПО на компьютер, подключенный к сети, проконсультируйтесь с сетевым администратором.

Дефектные диски. В маловероятном случае обнаружения дефектов на данном диске, пожалуйста, обращайтесь по адресу: disks@linuxformat.ru

Тираж изготовлен на Уральском электронном заводе, 620066, Россия, г. Екатеринбург, ул. Комвузовская 17-203, Лицензия МПТР России ВАФ № 77-15



Создание установочных дисков при помощи cdrecord

Самый быстрый способ записать ISO-образ на чистую матрицу – это *cdrecord*. Для всех перечисленных ниже действий потребуются права root. Для начала определите путь к вашему устройству для записи дисков. Наберите следующую команду:

```
cdrecord -scanbus
```

После этого на экране терминала должен отобразиться список устройств, подключенных к вашей системе. SCSI-адрес каждого устройства представляет собой три числа в левой колонке, например, 0,3,0. Теперь вы можете с легкостью записать образ на диск:

```
cdrecord dev=0,3,0 -v /путь к образу/image.iso
```

Чтобы упростить дальнейшее использование *cdrecord*, сохраните некоторые настройки в файле **/etc/default/cdrecord**. Добавьте по одной строке для каждого устройства записи (вероятно, в вашей системе присутствует всего одно такое устройство):

```
Plextor= 0,3,0 12 16M
```

Первое слово в этой строке – это метка, затем, после адреса SCSI-устройства вы должны указать скорость и размер буфера. Теперь вы можете заменить SCSI-адрес в командной строке на выбранную вами метку. Все будет еще проще, если вы добавите следующее:

```
CDR_DEVICE=Plextor
```

Все, что вам теперь нужно для записи ISO-образа – это набрать команду

```
cdrecord -v /path/to/image.iso
```

Если вы не из числа любителей командной строки, в таком случае вам придет на помощь утилита *gcombust*. Запустите ее из-под root, выберите вкладку Burn и ISO 9660 Image в верхней части окна. Введите путь к образу, который вы хотите записать на диск, и смело нажимайте на Combust!. Пока ваш образ пишется на диск, можете выпить чашечку кофе.

Другая ОС?

Вам не обязательно использовать Linux для записи компакт-диска. Все необходимые файлы уже включены в ISO-образ. Программы вроде *cdrecord* просто переносят данные на чистую матрицу. Если у вас нет устройства для записи дисков, можно найти того, у кого оно есть, и записать диск на его компьютере. На нем может стоять Windows, Mac OS X, AmigaOS, или любая другая ОС.

Нет устройства для записи дисков?

А что, если у вас нет устройства, с помощью которого можно было записать образ на диск? Вы знаете кого-либо с таким устройством? Вам не обязательно использовать Linux для записи дисков, подойдет любая операционная система, способная распознать пишущий привод (см. выше).

Некоторые дистрибутивы умеют монтировать образы дисков и выполнять сетевую установку или даже установку с раздела жесткого диска. Конкретные методы, конечно, зависят от дистрибутива. За дополнительной информацией обращайтесь на web-сайт его разработчика. LXF

Дистрибутив Linux

Mandriva Free 2007

Быстрый и простой инсталлятор? Есть! Отличное распознавание оборудования? Есть! Масса прекрасных программ? Тоже есть, и все это объединено в прекрасном рабочем столе KDE. У Mandriva 2007 проставлены галочки во всех нужных графах хорошего настольного дистрибутива, поэтому мы особенно рады представить версию Free – окончательный релиз, а не наскоро сляпанный релиз-кандидат – на нашем DVD.



› **У Mandriva превосходен не только набор программ, но и сообщество пользователей.**

Вместе с обновлением обычных программ (ядро 2.6.17.5, *X.org 7.1*, KDE 3.5.4 и Gnome 2.16), Mandriva 2007 – первая версия дистрибутива, использующая трехмерные рабочие столы *AIGLX* и *Xgl*. Перевернете страницу – узнаете больше.

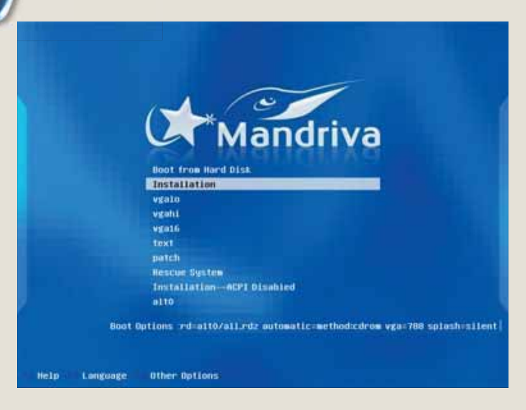
Для гладкой работы Mandriva нужен x86 ПК не менее чем на 1 Ггц, с 256 МБ ОЗУ и 5 ГБ пространства на жестком диске. (Может быть, вам и удастся добиться чего-то на более слабой машине, например, при 128 МБ ОЗУ, но тогда не ждите особой скорости.) Чтобы установить Mandriva, загрузите компьютер с LXFDVD и в появившемся меню выберите Install. Если вы не видите меню загрузки, то, возможно, ваш компьютер не настроен на загрузку с DVD. Так же, как и при установке любой новой ОС, не помешает сделать резервные копии данных, особенно если вы собираетесь установить на компьютере Linux вместе с Windows! Обратите внимание – наша DVD-версия Mandriva скопирована с набора из нескольких CD, и если программа установки в какой-то момент предложит вставить другой диск, просто задвиньте назад DVD-лоток и нажмите OK для продолжения установки. На



› **При интегрированных (т.е. беспроблемных!) *AIGLX* и *Xgl*, крутые 3D-рабочие столы доступны за пару щелчков мыши.**

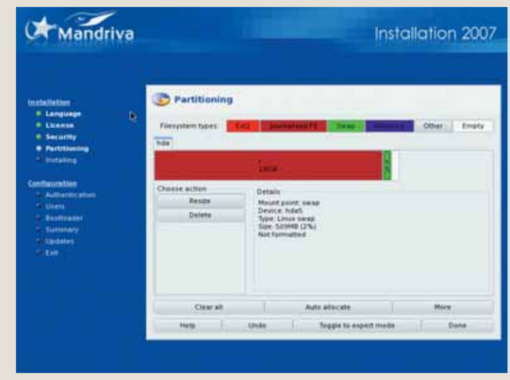
LXFDVD собрано все, что нужно: при желании вы сможете создать версию Mandriva 2007 на 4 CD для машин только с CD-приводом – см. **index.html** на DVD, там вы найдете подробную информацию. Если во время установки у вас возникнут проблемы, посетите наш форум: www.linuxforum.ru. »

Шаг за шагом: Установка Mandriva Free 2007



1 Загрузка

Загрузите ПК с LXFDVD и нажмите кнопку «стрелка вниз» и Enter для выбора пункта Installation. Если при загрузке возникли проблемы, попробуйте 'vga16' или отключение ACPI.



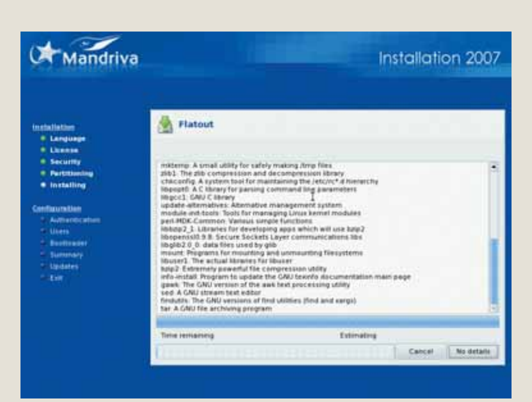
2 Запуск

Следуйте инструкциям программы установки; при разбиении диска на разделы, можете выбрать весь диск или откусить чуть-чуть у Windows (рекомендуем минимум 5 ГБ для корневого раздела и 512 МБ для раздела подкачки).



3 Приложения

При появлении этого экрана выберите установку по умолчанию, если не уверены, что можете сделать лучше. Если у вас потребуют поменять CD, просто снова вставьте LXFDVD и продолжайте установку.



4 Финал

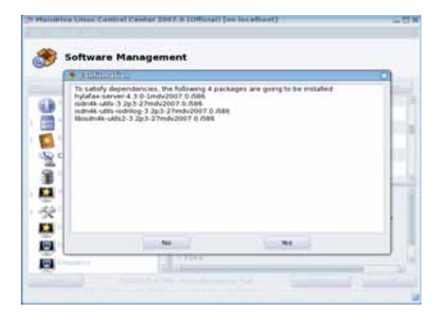
Скопировав файлы, займитесь настройкой. Обратите внимание на ваш пароль администратора и обычные пароли пользователей. После завершения, извлеките DVD и перезагрузитесь в Mandriva!

» Вероятно, вам не терпится опробовать новые функции, особенно трехмерный рабочий стол. Так это просто: нажмите Star Menu > System > Configuration > Configure Your Computer, и по запросу введите пароль администратора; вас допустят в Центр управления Mandriva (Mandriva Control Center). Это главный пункт настроек Mandriva, от загрузки и управления оборудованием до установки программ и сетевых соединений.

Настройка оборудования

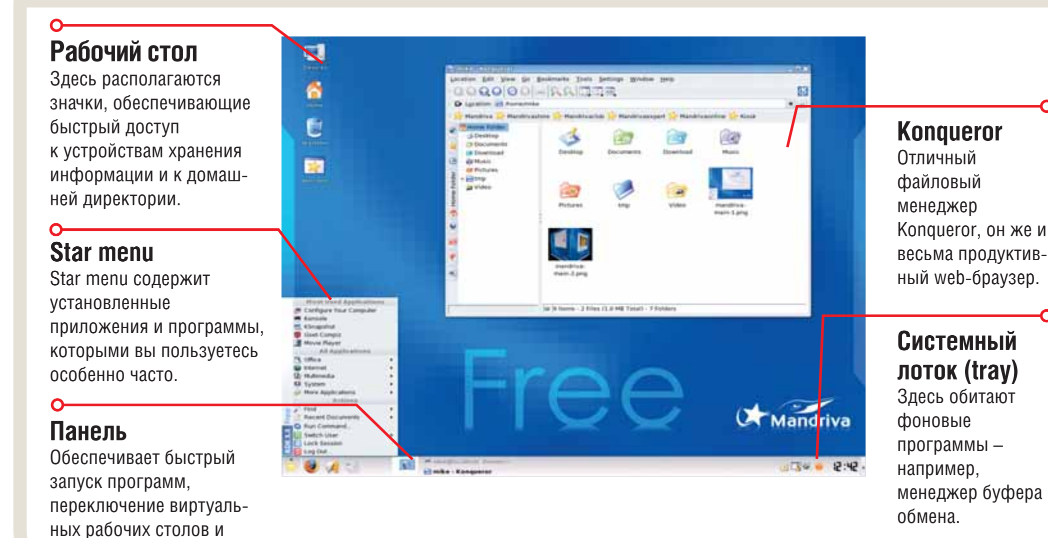
Нажмите вкладку Hardware слева, затем – кнопку Configure 3D Desktop Effects (Настройка эффектов 3D-рабочего стола) справа. Затем можете выбрать Все эффекты 3D-рабочего стола (Full 3D Desktop Effects) и ваш тип графического сервера – мы рекомендуем *AIGLX*. Сделав свой выбор, нажмите OK, и Mandriva предложит установить некоторые пакеты с диска; по запросу, вставьте LXFDVD. После копирования файлов, компьютер попросит вас перезапустить X-сервер – это можно сделать, нажав Ctrl+Alt+Backspace. Вы вернетесь в окно входа в систему; вводите ваши имя пользователя и пароль – готово!

Вернетесь-то вы в обычное окружение, но теперь можно начинать эксперименты с умопомрачительными эффектами трехмерного рабочего стола-кубика. Нажмите клавиши Ctrl+Alt+стрелка вправо для переключения виртуальных рабочих столов – и наблюдайте за кубом! Попробуйте придержать Ctrl+Alt, затем щелкнуть левой кнопкой мыши на рабочем столе, и плавно повращать куб. Еще, при нескольких открытых окнах, переместите курсор мыши в верхний правый угол экрана, и вы увидите все ваши окна в размере с ноготок, *а-ля Expose* в Mac OS X. Побольше о свойствах трехмерного рабочего стола можно узнать, запустив **gset-compiz** из пункта Run в Star menu.



› **Инструмент gset-compiz поможет настроить ваш трехмерный рабочий стол.**

Знакомство с рабочим столом Mandriva



Рабочий стол
Здесь располагаются значки, обеспечивающие быстрый доступ к устройствам хранения информации и к домашней директории.

Star menu
Star menu содержит установленные приложения и программы, которыми вы пользуетесь особенно часто.

Панель
Обеспечивает быстрый запуск программ, переключение виртуальных рабочих столов и панель задач.

Konqueror
Отличный файловый менеджер Konqueror, он же и весьма продуктивный web-браузер.

Системный лоток (tray)
Здесь обитают фоновые программы – например, менеджер буфера обмена.

Дополнительное ПО

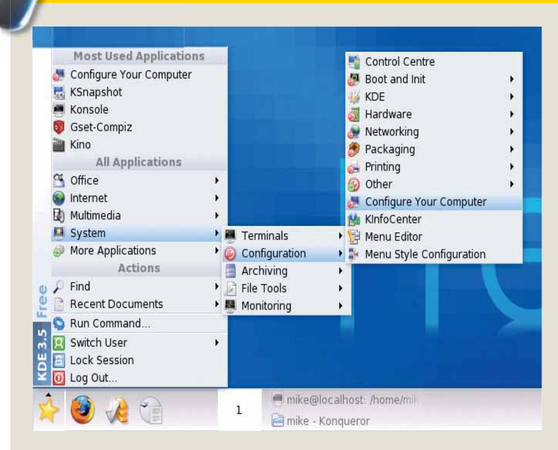
Даже дебютантам в Linux рабочий стол должен быть понятен и удобен: внизу – панель задач, а Star menu открывает список приложений. Вы можете настроить свой компьютер и внешний вид рабочего стола через Mandriva Control Center, как было описано выше.

Возможно, вам захочется добавить еще какие-нибудь программы с DVD. Откройте Mandriva Control Center, если потребуется, введите пароль администратора, затем щелкните по зеленому плюсу. Теперь можете щелкать по категориям программ слева и выбирать соответствующие приложения справа. Сделав выбор, щелкните Apply, и Mandriva попросит вас вставить диски с программами; просто вставьте LXFDVD. Затем начнется установка вашего нового ПО.

Обратите внимание, что в силу лицензионных соглашений и условий распространения, Mandriva не поставляет программы, являющиеся проприетарными – например, драйверы видеокарт ATI/Nvidia и медиа-кодеки. Однако, вы можете легко получить доступ к ним из Mandriva, посетив сайт http://easyurpmi.zarb.org и проследовав всем инструкциям. Выбрав 2007 Official, проверьте наличие пяти флажков Package Source (включая plf-free и plf-nonfree), затем выберите для каждого репозитория зеркало, ближайшее к вашей стране. Вам будет предоставлен список команд, которые вы можете вставить в окно терминала в качестве администратора, чтобы получать доступ к свежим пакетам по мере их обновления.

» Для подробной информации и поддержки Mandriva посетите www.mandriva.com/en/linux/2007 и www.mandrivausers.org, там же вы сможете оставить сообщения на форуме и получить помощь у пользователей Mandriva.

Шаг за шагом: Установка новых программ



1 Menu
Нажмите Star Menu >System > Configuration > Configure Your Computer для запуска Mandriva Control Center.



2 Пакеты
Для установки пакетов, щелкните по зеленому плюсу. Появится это окно; категории программ, которые вы хотите установить, выбираются слева.

Дистрибутив Linux

Fedora Core 6

Помимо стандартного обновления пакетов, Fedora Core 6 включает по-настоящему крутые штуки: *AIGLX* и интеграцию с *Xen*. Первое означает не менее трехмерный рабочий стол, чем в Mandriva 2007 и не далее, чем на расстоянии щелчка мышью. Самостоятельная установка *Compiz* может вызывать затруднения – но только не в Fedora Core! Просто добавьте к вашей инсталляции пакет *Compiz* и включите пункт «Desktop Effects» в диалоге Preferences. Готово!

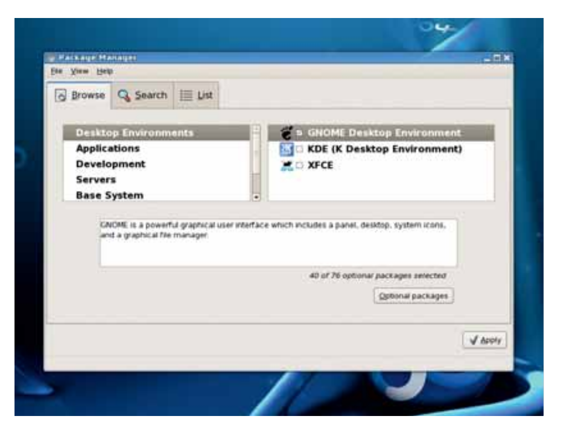
Интеграция с *Xen* придется по вкусу истинным хакерам, предпочитающим внешнему лоску богатое содержание. Виртуальные машины теперь можно настраивать через интерфейс Virtual Machine Manager! Если вы когда-либо пытались одолеть *Xen* в одиночку, то по достоинству оцените этого помощника.

Разработчики включили в Fedora Core 6 внушительную подборку программ – в том числе, Gnome 2.16 и KDE 3.5.4. Благодаря изменениям в подсистеме динамической компоновки, приложения стали запускаться немного быстрее. Определенной доводке подверглась и файловая система ext3, используемая в Fedora Core по умолчанию.



› **Миниатюры запущенных программ в стиле OS X – спасибо, *Compiz*!**

Чтобы установить Fedora Core 6, просто загрузитесь со второй стороны нашего DVD. Для работы с разумной скоростью потребуется ПК с процессором не ниже 1 ГГц, 512 МБ оперативной памяти и теми же 5 ГБ на жестком диске – да, не забудьте сделать резервную копию важных данных! Вы можете превратить DVD в набор из 5CD – просто следуйте инструкциям, которые найдете в файле **index.html** на диске. В случае возникновения проблем обращайтесь на форум – www.linuxforum.ru.



› ***Pirut* упрощает установку ПО, хотя может работать не слишком расторопно, если используется *Yum*.**

3 Установка

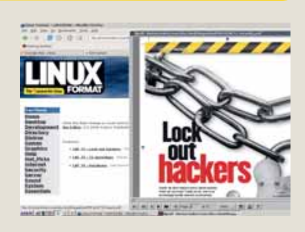
Поставьте флажки рядом с названиями программ, которые вы хотите установить, затем нажмите Apply, а когда вас попросят вставить следующий диск, вставьте LXF DVD. Все, готово!

Документация

Материалы LXF в формате PDF

В разделе Журнал/PDFs на DVD этого месяца вы найдете три обширных материала из предыдущих выпусков LXF в формате PDF. Просмотрщик PDF является составной частью большинства рабочих столов Linux (например, в Gnome есть *Evince*, а в KDE – KPDF), но если вы не можете найти такой программы в вашем дистрибутиве, установите старый добрый *Xpdf* из раздела **Рабочий стол** нашего DVD. Не забудьте, что PDF-файлы имеют высокое разрешение, и на старых машинах (менее 1 ГГц) могут загружаться в течение нескольких секунд.

Наш первый материал, из LXF71, фокусируется на проблеме непреходящей важности – на безопасности. Какими бы новыми ни были программы на вашем компьютере, всегда найдутся вредоносные программы еще новее, и вместе с тем – новые технологии, позволяющие отогнать вредителей. Мы начали с общего обзора рисков, с которыми сталкиваются обычные пользователи Linux, а затем перешли к настройке брандмауэра, защите системы с помощью *Bastille*, сканированию уязвимостей в *Nessus* и шифрованию. Это – полезное чтение, если вы работаете через Linux с сетью; плюс к тому, вы разберетесь с большинством инструментов, упомянутых в разделе **Безопасность** нашего диска.



› **Создайте личную библиотеку документации Linux.**

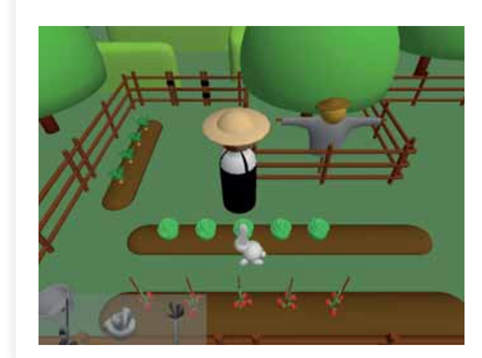
Теперь о втором материале: «72-я скорость». Название говорит само за себя, и оно правдиво: в нем 72 совета и подсказки, как разогнать вашу машину. Чтобы облегчить восприятие, мы распределили советы по темам: загрузка, рабочий стол, приложения, базы данных, сервер, ядро и оборудование.

И, наконец, есть обширное руководство из LXF76, посвященное настройке оборудования в Linux. Если вы хоть раз героически сражались с каким-нибудь устройством, заставляя его работать с вашим дистрибутивом, читайте: тут говорится о клавиатурах, мышах, джойстиках, принтерах, камерах, сканерах, видеокартах, сетевых соединениях и т.д. Приятного времяпровождения!

И наконец...

Пара слов о других материалах нашего DVD. Для разработчиков мы включили последние версии Mono и *MonoDevelop*, которые вы сможете применить, ознакомившись с нашим руководством на стр. 56.

А еще вас ждут: подборка новых приложений рабочего стола, интернет-инструменты и многое другое. Чтобы развлечься, загляните в раздел Игры: *Bouncy* – идеальная игра для детей, в ней кролик норовит сожрать урожай овощей, выращенных фермером. Чтобы прыгнуть, используйте клавиши управления курсором, чтобы слопать ближайший овощ – пробел, а чтоб зарыться под землю – D. Прыг-скок! LXF



› ***Bouncy*: Кролик прыг – кролик скок – лишь бы фермер не засек!**



Журнал зарегистрирован Федеральной службой по надзору за соблюдением законодательства в сфере массовых коммуникаций и охране культурного наследия
ПИ № ФС77-21973 от 14 сентября 2005 года
Выходит ежемесячно. Тираж 5000 экз.

РЕДАКЦИЯ РУССКОЯЗЫЧНОЙ ВЕРСИИ:
ГЛАВНЫЙ РЕДАКТОР
Валентин Синицын info@linuxformat.ru
Литературные редакторы
Родион Водейко, Иван Мищенко, Елена Толстякова
Переводчики
Александр Бикмеев, Светлана Кривошеина, Александр Кузьменков, Алексей Опарин, Валентин Разважаев, Сергей Супрунов, Александр Черных, Юлия Шабунио
Допечатная подготовка
Мария Пучкова, Родион Водейко
Креативный директор
Станислав Медведев
Технический директор
Денис Филиппов
Директор по рекламе
Денис Игнатов +7 812 965 7236 advert@linuxformat.ru
Заместитель генерального директора
Софья Виниченко
Генеральный директор
Павел Фролов
УЧРЕДИТЕЛИ
частные лица
ИЗДАТЕЛИ
Станислав Медведев, Павел Фролов

Отпечатано в типографии «Текст», ООО «ППК «Текст»
188680, Ленинградская область, Всеволожский район, Колтуши, д.32
Заказ _____
Пре–пресс: *d.r.i.v.e-group*

РЕДАКЦИЯ АНГЛОЯЗЫЧНОЙ ВЕРСИИ:
Редактор Ник Вейч (Nick Veitch) nick.veitch@futurenet.co.uk
Заместитель редактора Пол Хадсон (Paul Hudson) paul.hudson@futurenet.co.uk
Старший художественный редактор Мартин Парфитт (Martin Parfitt) mparfitt @futurenet.co.uk
Художественный редактор Эфрейн Эрнандес-Мендоса (Efrain Hernandez-Mendoza) efrain.hernandez-mendoza@futurenet.co.uk
Новостной редактор Майк Сондерс (Mike Saunders) mike.saunders@futurenet.co.uk
Литературный редактор
Ребекка Смолли (Rebecca Smalley) rebecca.smalley@futurenet.co.uk
Штатный автор
Грэм Моррисон (Graham Morrison) graham.morrison@futurenet.co.uk
Ассистент по выпуску
Эндрю Грегори (Andrew Gregory) andrew.gregory@futurenet.co.uk

Авторы
Ладислав Боднар (Ladislav Bodnar), Нейл Ботвик (Neil Bothwick), Д-р Крис Браун (Dr. Chris Brown), Энди Ченнел (Andy Channelle), Эди Хадсон (Andy Hudson), Дэниэль Кингшот (Daniel Kingshott), Брайан Суда (Brian Suda), Евгений Балдин, Андрей Боровский, Андрей Прахов, Петр Семилетов, Александр Супрунов, Алексей Федорчук, Антон Черноусов, Илья Шпаньков
Художественные ассистенты: Алекс Дюс (Alex Duce), Марк Митчелл (Mark Mitchell), Элвин Витмен (Alvin Weetman)
Фотографии: Дэвид Блэнкенхорн (David Blankenhorn), Джейсон Каплан (Jason Kaplan)
Иллюстрации: Нейл Бартлет (Neil Bartlett), Пол Блехфорд (Paul Blachford), Elly Walton Illustrations, Крис Винн (Chris Winn)

КОНТАКТНАЯ ИНФОРМАЦИЯ
UK: Linux Format, 30 Monmouth Street, Bath BA1 2BW
Tel 01225 442244 Email: linuxformat@futurenet.co.uk
РОССИЯ:
Санкт-Петербург (редакция): ул. Гончарная, 23, офис 54, телефон: (812) 717-00-37
Представительство в Москве:
пр.Мира, 161, телефон +7(495) 799-18-63, +7(495)136-88-45
Email: info@linuxformat.ru, Web: www.linuxformat.ru



Встречайте



Узнайте, почему Mono принимает и продвигает Microsoft .NET как технологию рабочего стола Linux будущего!

Приближается Python 3000...

Гвидо ван Россум обновляет Python – хотите знать, что изменится?



Майкл Тиман

Вице-президент Red Hat – о соглашении между Novell и Microsoft.



Внимание! Содержание следующих выпусков может изменяться без уведомления

ПОДПИСКА НА LINUX FORMAT

ПОДПИСКА В ЛИНУКСЦЕНТРЕ

Сколько стоит подписка?

Подписка на журнал «Linux Format» 12 номеров (январь, февраль, март, апрель, май, июнь, июль, август, сентябрь, октябрь, ноябрь, декабрь) стоит 1800 рублей
Подписка на журнал «Linux Format» 6 номеров (июль, август, сентябрь, октябрь, ноябрь, декабрь 2006 года) стоит 900 рублей

Как оформить подписку?

Чтобы оформить подписку на журнал «Linux Format», необходимо зарегистрироваться в интернет-магазине Linuxcenter.Ru, указав ФИО и подробный почтовый адрес подписчика, заказать товар «Подписка на журнал «Linux Format» 12 номеров 2006 года», или товар «Подписка на журнал «Linux Format» второе полугодие 2006 года», получить от системы квитанцию для оплаты в любом отделении Сбербанка (для физических лиц) или счет для оплаты по безналичному расчету (для юридических лиц)

Как оплатить подписку?

– по выставленному счету (для юридических лиц)
– по квитанции в любом отделении Сбербанка

Плюсы подписки

- подписка дешевле!
- гарантированное получение нового номера журнала!

ПОДПИСКА - 2007!

ПОДПИСКА ПО КАТАЛОГАМ РФ

Каталог агентства «РОСПЕЧАТЬ» – подписной индекс 20882

Каталог «ПРЕССА РОССИИ» – подписной индекс 87974

Ф. СП-1

Министерство связи РФ

АБОНЕМЕНТ НА ЖУРНАЛ

Linux Format

ИНДЕКС ИЗДАНИЯ

КОЛИЧЕСТВО КОМПЛЕКТОВ

НА 2007 ГОД ПО МЕСЯЦАМ

1	2	3	4	5	6	7	8	9	10	11	12

КУДА ______ ПОЧТОВЫЙ ИНДЕКС ______ АДРЕС ДОСТАВКИ

КОМУ ______ ФАМИЛИЯ, ИНИЦИАЛЫ

ДОСТАВОЧНАЯ КАРТОЧКА

ПВ	МЕСТО	ЛИТЕР

НА ЖУРНАЛ ______ ИНДЕКС ИЗДАНИЯ

Linux Format

НАИМЕНОВАНИЕ ИЗДАНИЯ

СТОИ-МОСТЬ	ПОДПИСКИ	РУБ. КОП.	КОЛИЧЕСТВО КОМПЛЕКТОВ	
	ПЕРЕАДРЕСАЦИИ	РУБ. КОП.		

НА 2007 ГОД ПО МЕСЯЦАМ

1	2	3	4	5	6	7	8	9	10	11	12

КУДА ______ ПОЧТОВЫЙ ИНДЕКС ______ АДРЕС ДОСТАВКИ

КОМУ ______ ФАМИЛИЯ, ИНИЦИАЛЫ



ПОДПИСКА НА LINUX FORMAT

ПОДПИСКА ПО КАТАЛОГАМ СНГ И БЛИЖНЕГО ЗАРУБЕЖЬЯ

Каталог «Российская Пресса» – совместный проект Государственного предприятия «Казпочта», Агентства «Книга-Сервис» и АРЗИ.

Блок изданий АРЗИ в национальных Каталогах Украины и Беларуси. В Азербайджане, Армении, Грузии, Киргизии, Узбекистане и Молдове - по изданиям, включенным в Объединенный каталог, распространяемые через АРЗИ.

Азербайджан
- по Объединенному каталогу российских изданий через Предприятие по распространению печати «Гасид»(370102, г. Баку, ул. Джавадхана, 21);

Армения
- по списку номенклатуры «АРЗИ» через ГЗАО «Армпечать» (375005, г.Ереван, пл.Сасунци Давида, д.2) и ЗАО «Контакт-Мамул» (375002, Г.Ереван, ул.Сарьяна, 22);

Белоруссия
- по Каталогу изданий стран СНГ через РГО «Белпочта» (220050, г.Минск, пр-т Ф.Скорины, 10);

Грузия
- по списку номенклатуры «АРЗИ»» через АО «Сакпресса» (380019, г.Тбилиси, ул.Хошараульская, 29) и АО «Мацне» (380060, г.Тбилиси, пр-т Гамсахурдия, 42);

Казахстан
- по Каталогу «Российская Пресса» через ОАО «Казпочта» и ЗАО «Евразия пресс»;

Молдавия
- по каталогу через ГП «Пошта Молдавей» (МД-2012, г.Кишинев, бул.Штефан чел Маре, 134);
- по списку через ГУП «Почта Приднестровья» (MD-3300, г.Тирасполь, ул.Ленина, 17);
- по прайс-листу через ООО Агентство «Editil Periodice» (2012, г.Кишинев, бул. Штефан чел Маре, 134).

Узбекистан
- по Каталогу «Davriy nashrlar» российские издания через Агентство по распространению печати «Davriy nashrlar» (7000029, Ташкент, пл.Мустакиллик, 5/3, офис 33);

Украина
- Киевский главпочтамт.
- Подписное агентство «KSS» Телефон/факс (044)270-62-20, 270-62-22

ПОДПИСКА НА LINUX FORMAT

Агентство “Centerpress”

Сколько стоит подписка?
Подписка на журнал "Linux Format" 12 номеров (январь, февраль, март, апрель, май, июнь, июль, август, сентябрь, октябрь, ноябрь, декабрь 2007 года) стоит 1800 рублей.

Как оформить подписку?
Чтобы оформить подписку на журнал "Linux Format", необходимо зарегистрироваться в интернет-агентстве Centerpress.ru, указав ФИО и подробный почтовый адрес подписчика, заказать товар "Подписка на журнал "Linux Format" на 2007 год 12 номеров (01-12 / 2007)", получить от системы квитанцию для оплаты в любом отделении Сбербанка (для физических лиц) или счет для оплаты по безналичному расчету (для юридических лиц)

Агентство “Centerpress” : www.centerpress.ru

Все Плюсы подписки!
- Подписка дешевле!
- Гарантированное получение журнала!

По каталогам РФ

Каталог агентства “РОСПЕЧАТЬ” - подписной индекс

20882

Каталог “ПРЕССА РОССИИ” - подписной индекс

87974



АЛЬТЕРНАТИВНЫЕ АГЕНСТВА РФ

Агентство «Интер-Почта» (095) 500-00-60, курьерская доставка по Москве.

Агентство «Вся Пресса» (095) 787-34-47

Агентство «УралПресс»

- Екатеринбург, Березовский, В. Пышма, Первоуральск
тел. (343) 375-80-71, 375-84-93, 375-84-39, факс 375-62-74, info@ural-press.ru
- Нижний Тагил
тел. (3435) 411448, 417709, ntagil@ural-press.ru
- Челябинск
тел. (351) 262-90-03, 262-90-05, pochta@chel.surnet.ru
- Пермь
тел. (3422) 60-24-40, 60-22-95, 60-35-42, parma-press@permonline.ru